ENSURING ORACLE® DATA INTEGRITY WITH NETAPP SNAPVALIDATOR™

Tushar Patel, Fred Schimscheimer, Eyal Traitel, Jai Desai & Sunny Ng, Network Appliance, Inc.
December 2006 | TR-3414

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Organizations of any size that rely heavily on Oracle data integrity need a guaranteed safeguard against data corruption. Because database servers and storage devices reside at opposite ends of the I/O path, each data block transfer passes through a series of logical layers involving hardware and software from multiple vendors. Other factors, such as application anomalies and human error, present additional risk. As a result, data corruption can occur at every stage of the process, even with the protection inherent in the most robust storage systems. The cumulative impact of these corruptions can cause considerable disruption to business continuity, which can be time-consuming and costly to resolve.

In today's data-critical business environments, customers want to:

- Detect and prevent potential data corruption before it happens
- Prevent accidental overwrites of business-critical Oracle data
- Detect any data corruption introduced during backups or data lifecycle management

NetApp SnapValidator addresses all the needs listed above and provides the highest possible level of protection for Oracle data. It detects and prevents potential corruptions of Oracle data before they happen. By adding intelligence and database awareness to modular storage systems—across SAN/iSAN and NAS protocols—NetApp extends the advantages of checksum functionality to a greater variety of organizations. SnapValidator is tightly integrated with the Oracle Database architecture and complies with the Oracle Hardware Assisted Resilient Data (HARD) initiative. When initiating data transfers, the Oracle instance assigns a unique value to each write request based on the sum of the data and sends the data set down the I/O path to the NetApp storage system. Upon receipt, NetApp SnapValidator calculates the sum of the data and compares it to the Oracle value. If these values match, the system writes the information to disk and reports a successful write. If the identifiers do not match, the system prevents the data corruption and produces an alert that offers options for resolving the problem immediately to prevent further problems. You gain the assurance that every bit of data is written with 100% reliability, and you avoid the cost of unplanned disruption and downtime associated with resolving data corruptions.

As good as this concept sounds; it comes at an extra cost, which is extra storage resource utilization. It is also obvious that the cost of running SnapValidator can significantly vary depending upon the type of workload, particularly by number of writes. This paper highlights the cost of running SnapValidator for a database workload with a mix of read/write and random/sequential transactions.

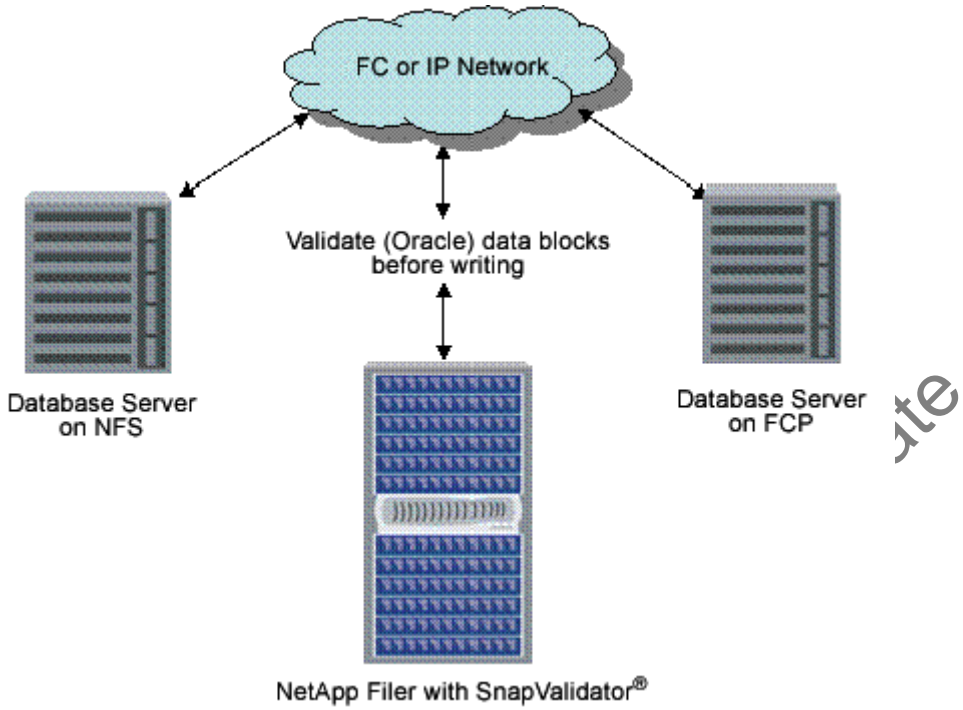
WHAT IS SNAPVALIDATOR?

SnapValidator is the NetApp implementation of the Oracle HARD initiative, a comprehensive program designed to prevent data corruptions before they happen. Data corruptions can have a catastrophic effect on a database and eventually on critical business operations. By implementing Oracle data validation algorithms inside storage devices, Oracle will prevent corrupted data from being written to permanent storage. This type of end-to-end, high-level software to low-level hardware validation has never been implemented before. HARD eliminates a large class of failures that the database industry has so far been powerless to prevent. RAID has gained a wide following in the storage industry by ensuring the physical protection of data, HARD takes data protection to the next level by going beyond protecting physical bits to protecting business data.

NetApp SnapValidator is tightly integrated with the Oracle Database architecture and complies with the Oracle HARD initiative. By providing end-to-end checksum functionality, NetApp SnapValidator assures that the data corruption and errors can be detected and prevented before they propagate in the system and incur liability. NetApp SnapValidator key benefits are:

- Increased data reliability via end-to-end data protection
- Improved operational efficiency and avoided costs
- Modular accessibility
- Multiprotocol support

HOW DOES SNAPVALIDATOR WORK?



SnapValidator for Oracle (SVO) is a Data ONTAP™ WAFL® enhancement that allows the NetApp filer to validate Oracle data when it is being written to the filer. This allows the storage to stop data corruptions caused by the host operating system, network, and other intermediaries from making it onto disk. As mentioned earlier, NetApp SnapValidator is tightly integrated with the Oracle Database architecture. Following is a step-by-step breakdown of how a write operation is performed from Oracle instance to NetApp storage with SnapValidator.

1. At the time of transaction commit, Oracle creates checksums related to the data that it is writing to disk and ships it along with the original data.
2. Data traverses through the host OS, storage stack, and storage network and finally reaches the storage.
3. Before the write is committed, the NetApp storage appliance performs the following:
    a. Checksum and verification against the data that was sent
    b. Verification of Oracle block number against the block the data is getting written into
    c. Verification of length of write and the intended write size
    d. Oracle magic number check
4. If all of the verification goes through fine, the NetApp storage appliance writes the data to disk; if not, it sends an error back to the database. SnapValidator can be preconfigured about how to handle cases where the test fails. In addition, SnapValidator always logs failed checks.

Here are some noteworthy points about SnapValidator:

- SVO cannot mix Oracle data and data not from Oracle in the same volume.
- SVO is enabled at a volume level and works on both flexible and traditional volumes.
- SVO cannot be used on the root volume of the filer.
- SVO supports both Oracle9i and Oracle 10g.

The next section highlights generic commands and setup options for SnapValidator.

SNAPVALIDATOR SETUP

SnapValidator is very easy to set up on the filer for both a new database and an existing configuration.

Here are the steps to configure SnapValidator for a new database.

1. Create a volume for Oracle data.
2. Set up Oracle, build databases, and test environment.
3. License SnapValidator.
4. Enable SVO and checksums.
5. Set svo_offset on LUNs if using LUNs. This is very important for proper operation. Please see note below about how to set svo_offset for LUNs.
6. Test again to make sure there are no false errors.
7. Enable svo_reject_errors.
8. Put database into production.

Here are the steps to enable SnapValidator on an existing configuration.

1. Reorganize data so that Oracle9*i* R2 data is on its own volume. This means only Oracle data; no config files, no scripts, and no text files of any kind on that volume.
2. If using LUNs, set the svo_offset. This is very important for proper operation. Please see note below about how to set svo_offset for LUNs.
3. Enable SVO and svo_checksum.
4. Test for false errors and setup problems.
5. Enable svo_reject_errors.
6. Put database into production.

How to set up LUN offset correctly

First, enable SVO on the filer with checksum. Make sure to keep svo_reject_errors off and svo_allow_rman off. Next, set the LUN's SVO offset to 0 as indicated below:

lun set svo_offset /vol/test/c1 0

Since the LUN offset varies from OS to OS and also from volume managers (LVMs), we need to find out correct offset via dd command as indicated below:

host% dd if=/dev/zero bs=512 count=1 of=/dev/rsvo_vol_2

It should generate an error that would look like:

Validation error Zero Data:: v:9r2 vol:test inode:104 length:512 Offset: 2228224

Now, set the correct offset to LUN as:

lun set svo_offset /vol/test/c1 2228224

The appendix summarizes more details about SnapValidator commands and options.

SNAPVALIDATOR PERFORMANCE IMPACT ON FILER

Up to this point, we have talked about SnapValidator capabilities and the great protection and peace of mind that it offers. However, all of that comes at some cost in terms additional filer resource consumption. Obviously, checksum calculation, data validations, and so on do consume extra CPU cycles and ultimately may affect I/O throughput in a heavily stressed filer. It is a fact and yet very tempting to say that cost is minimal and can vary depending upon number of writes. In this section, we attempt to address two most common questions:

1. What is the cost of running SnapValidator in terms of storage resource consumption?

2. When should someone consider running SnapValidator?

Test Workload and Setup

Before we dive into results, it is important to have a brief overview about the workload and some assumptions that we have made for this experiment. The goal for this exercise is to find the cost (in terms of additional filer resources) of running SnapValidator in a realistic environment. We identified the following requirements for a workload that would meet our needs.

Workload Characteristics/Requirements

- The workload must be a database stress test that performs various transactions against the Oracle Database on the filer.
- It must be a representative of a realistic workload encountered in day-to-day operations.
- It must provide user-configurable mix of sequential/random and read/write I/O operations on the filer.
- It must be consistent but still have a large enough data set to facilitate random I/O operations.

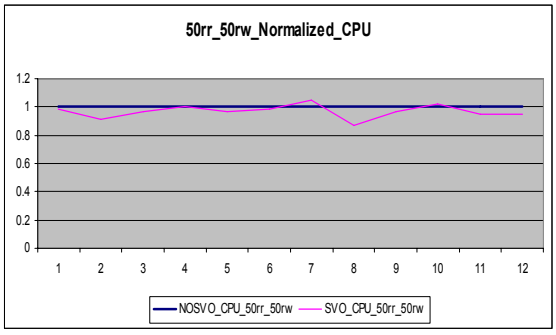
For this exercise, we decided to run Oracle Storlab workload since it addressed the requirements listed above. Storlab is a simple workload from Oracle that performs actual database transactions on a user-configurable mix of sequential/random and read/write operations. The database size is roughly 120GB, and the largest tablespace is about 39 million rows. Here are the setup assumptions we made for this exercise:

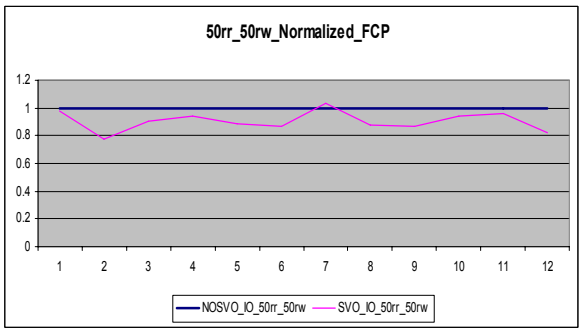
- The environment was not tuned to generate optimal results since the goal is to compare results with Oracle SVO on/off.
- We ran two types of I/O mix, which was most representative of daily business operations:
  - 50% random writes and 50% random reads
  - 25% random writes and 75% random reads
- For each run, all servers including storage subsystems were rebooted and database was restored.
- Storlab workload variances were minimized with multiple runs of each scenario.

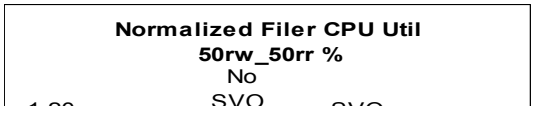
Results

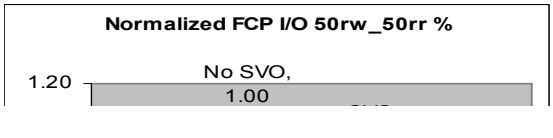
Here are the normalized results for Storlab tests we ran. We normalized the NOSVO cost for both CPU and FCP throughput to a constant 1. The charts below illustrate the resource utilization difference for SVO compared to non-SVO. It is also important to note that even though the workload was I/O intensive, the storage resources were not fully stressed during this exercise.

Run1: 50% Random Writes and 50% Random Reads





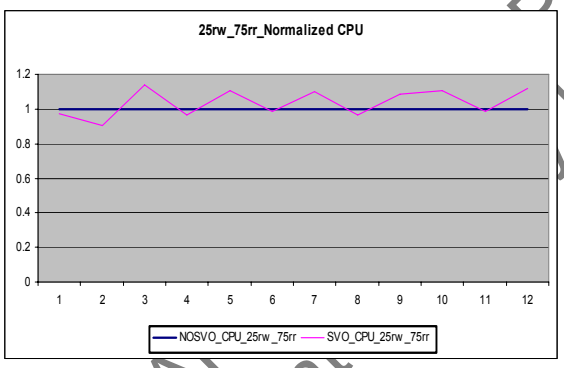


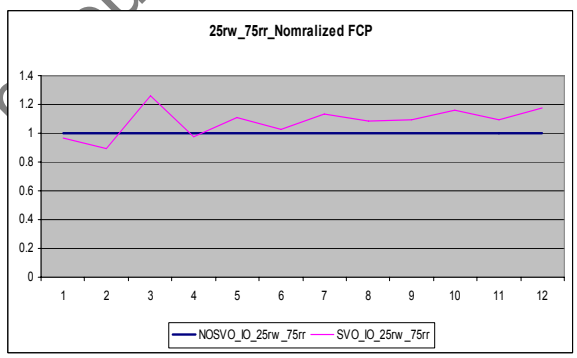


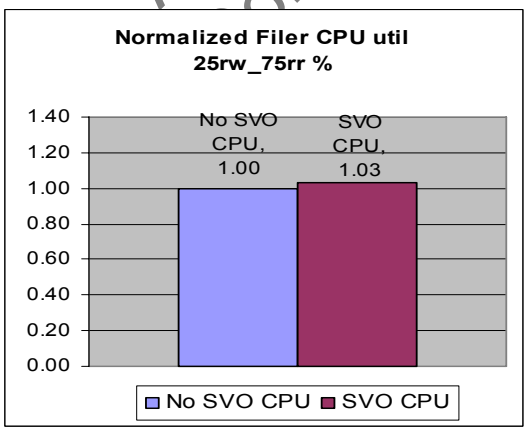
Observations

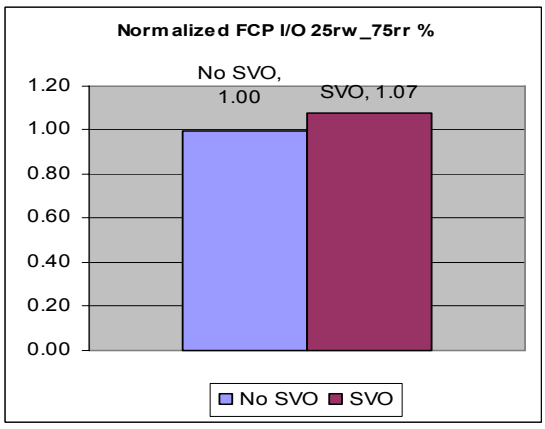
- For SVO, we observed about 3% less CPU utilization with about 9% less FCP traffic. These variations could mostly be due to variances in workload since there were no resource bandwidth constraints on the filer. In essence, the additional resource consumption for running SVO was negligible.

Run2: 25% Random Writes and 75% Random Reads









Observations

- For this run, the results were even more inconclusive. The tests with SVO did have a 3% higher CPU utilization but also did a little more work and had 7% higher FCP throughput. This is again mostly due to workload variations. Again, the cost of running SnapValidator was proved to be negligible for this test.

CONCLUSION

NetApp SnapValidator is tightly integrated with the Oracle Database architecture and complies with the Oracle HARD initiative. SnapValidator software provides the highest possible level of protection for Oracle data. It detects and rejects invalid Oracle data before it is written to the storage and hence prevents data corruptions caused by the host operating system, network, and other intermediaries from making it onto disk. SnapValidator is very easy to set up and is extremely flexible to allow various levels of Oracle data block checks (checksum, magic number, size, and so on). By adding intelligence and database awareness to modular storage systems, NetApp offers a comprehensive solution for data protection to a greater variety of organizations.

In terms of resource utilization, SnapValidator has a very low overhead that can be worth the cost for the benefits it offers. During our testing of Oracle Storlab workload, we concluded that SnapValidator consumed approximately 3% to 8% additional storage resources depending on the write vs. read ratios. Since SnapValidator resource utilization cost is directly proportional to number of writes, the OLTP workloads should have minimal impact on additional resource consumption.

APPENDIX

Additional References

- www.netapp.com/ftp/snapvalidator.pdf

  "SnapValidator Software for Oracle"

- www.oracle.com/technology/deploy/availability/htdocs/hardf.html

  "Oracle Hardware Assisted Resilient Data (HARD) Initiative"

SNAPVALIDATOR SYNTAX

Licensing SnapValidator

To license SnapValidator complete the following steps:

STEP	ACTION	

1	**Verify whether SnapValidator is licensed.** Enter the following command:
	`License`
	A list of all available services appears. Services that are enabled show the license code. Services that are not enabled are indicated as "not licensed." For example, the following line indicates that SnapValidator is not licensed:
	`SnapValidator not licensed`	

2	**If SnapValidator is...**	**Then...**

	Licensed	Proceed to Enabling SnapValidator Checks on Volumes
	Not licensed	Enter the following command: `license add license_code` *license_code* is the license code you received from NetApp when you purchased the SnapValidator license.

Enabling SnapValidator Checks on Volumes

You enable SnapValidator checks at the volume level. To enable SnapValidator checks on a volume, complete the following steps:

Note: You cannot enable SnapValidator on the root volume.

STEP	ACTION	
1	**On the filer command line, enable SnapValidator by entering the following command:** `vol options volume-name svo_enable on` All SnapValidator checks are enabled on the volume, with the exception of checksums.	
2	**If you want to...**	**Then enter the following command:**
	Enable data checksumming on the volume	`vol options volume-name svo_checksum on`
	Disable block number checks because the volume contains Oracle Recovery Manager (RMAN) backup data.	`vol options volume-name svo_allow_rman on`

	Set SnapValidator to reject all invalid operations and return an error log to the host and filer consoles.	`vol options volume-name svo_reject_errors on`
		If this option is not set to on, then SnapValidator detects invalid operations but only logs them as errors. The following shows a SnapValidator error example displayed on the filer console:
		`Thu May 20 08:57:08 GMT [filer_1: wafl.svo.checkFailed:error]: SnapValidator: Validation error Bad Block Number:: v:9r2 vol:flextest inode:98 length:512 Offset: 1298432`

3	**If the volume contains FCP LUNs, proceed to Enabling SnapValidator Checks on LUNs in the next section**.	

Enabling SnapValidator Checks on LUNs

If you enable SnapValidator on volumes that contain database LUNs, you must also enable SnapValidator checks on the LUNs by defining the offset to the Oracle data on each LUN. The offset separates the Oracle

data portion of the LUN from the host volume manager's disk label or partition information. The value for the offset depends on the operating system (OS) of the host accessing the data on the LUN. By defining the offset for each LUN, you ensure that SnapValidator does not check write operations to the disk label or partition areas as if they were Oracle write operations.

To define the disk offset for a LUN, complete the following steps:

STEP	ACTION
1	**Use the volume manager tools for your host OS to obtain the value of the offset.** For detailed information about obtaining the offset, see the vendor-supplied documentation for your volume manager.
2	**On the filer command line, enter the following command:** *lun set svo_offset* lun-path offset *Offset* is specified in bytes, with an optional multiplier suffix: c(1), w(2), b(512), k(1024), m(k*k), g(k*m), t(m*m). The offset specifies the number of bytes from the beginning of the LUN to the start of Oracle data.

Disabling SnapValidator on a Volume

To disable SnapValidator, complete the following steps:

STEP	ACTION
1	**On the filer command line, enter the following command:** *vol options* volume-name *svo_enable off* SnapValidator does not check Oracle write operations to files or LUNs. The settings for each type of check (for example, checksumming) are not disabled. If you reenable SnapValidator, the settings for each type of check are saved.
2	**To disable a specific SnapValidator option, enter the following command:** *vol options volume-name option off* *option* is one of the following: ▪ *svo_checksum* disables data checksumming on the volume. ▪ *svo_allow_rman* allows block number checks on the volume. You disable this option (set it to off) if the volume does not contain RMAN data. ▪ *svo_reject_errors* detects invalid operations but does not reject them. Invalid operations are only logged as errors.

Disabling SnapValidator Checks on a LUN

To disable SnapValidator checks on a LUN, complete the following step:

Step	Action
1	**On the filer command line, enter the following command:** `lun set lun-path svo_offset disable`

DISCLAIMER

Each environment has its own specific set of requirements, and no guarantees can be given that the results presented in this report will work as expected on other platforms. This paper should assist in the research and troubleshooting that may be required in a particular case and serve as a checklist of items to be aware of. Please forward any errors, omissions, differences, new discoveries, and comments about this paper to *tusharp@netapp.com*.