Technical Report

ParAccel on NetApp Proof of Concept

Jim Lanson, John Elliott, NetApp
Dan Flavin, ParAccel
August 2011 | TR-3951

ABSTRACT

In May 2011, ESG (Enterprise Strategy Group) published a lab validation report, titled ParAccel PADB and NetApp SAN Optimized Solution, based on a joint engagement with both NetApp and ParAccel. During that engagement, engineers from both companies demonstrated the superior functionality and rich feature set of the NetApp® SAN optimized solution for the ParAccel Analytic Database (PADB), which leverages ParAccel's patent-pending Blended Scan option. Many readers of that report may be interested in the details of how the tested configurations were actually set up. This document was written to provide that level of detail. It contains references to the ESG document and corresponding procedural information. We recommend that this document be read as a companion to the ESG report.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Companies today are constantly accumulating new data. Over the last five years, the average data warehouse size has grown by a factor of 10, with the typical data warehouse doubling in size every 6 to 9 months. Furthermore, companies are not only accumulating data at an unprecedented rate, but that data's scope, scale, and complexity continue to increase as well, compounding the challenges. The data being collected can also come from multiple sources, incorporate different data types such as image and voice, and include both structured and unstructured data.

The ability to quickly analyze and derive insights from your data even as the volume and complexity consistently increase provides both a competitive advantage and greater operational efficiency. Additionally, as the operational importance of data warehouses continues to increase, companies are now looking for holistic approaches that deliver better storage efficiency, speed, flexibility, and resiliency, without the trade-offs associated with traditional data warehouse technologies. For instance, as the number of data sources feeding data warehouses grows, you would need efficient solutions that can quickly undo a batch load if a load error is discovered. Wider use of the data across the organization also requires being able to quickly provide data access with the ability to spin up data marts in near-real time. Finally, as the size of data warehouses grows, customers are looking for the ability to maximize utilization and modularity, and to nondisruptively scale capacity as needed.

As a result, requirements for manageability, availability, and disaster recovery in data warehouse environments are reaching the same levels of importance as for transactional (OLTP) database systems. In addition to focusing on your data warehouse capacity, scalability, and performance needs—which are daunting enough by themselves—you now have to provide 24/7/365 availability, reduce the time needed for maintenance, and protect your systems against disaster, all without breaking your budget.

NetApp and ParAccel have partnered to allow our customers to meet the demands of today's complicated data warehouse environments by providing best-in-class analytical database and storage technology. ParAccel provides massively parallel processing. With its columnar storage architecture, only data that is pertinent to the executed query is read. ParAccel also provides high compression ratios, providing extremely high I/O throughput and efficiency. With NetApp's flexible volumes, storage can be provisioned quickly and easily. NetApp Snapshot™ and SnapRestore® features provide fast backup and recovery and NetApp FlexClone® enables fast creation of instant data marts that use very little actual storage space. Together NetApp unified storage and the ParAccel Analytic Database (PADB) provide all the features needed to meet the challenges listed above.

1.1 PURPOSE

In May 2011, ESG published the lab validation report "ParAccel PADB and NetApp SAN Optimized Solution": http://www.enterprisestrategygroup.com/media/wordpress/2011/05/ESG-Lab-Validation-Report-NetApp-ParAccel-May-11.pdf.

That report provides the following:

- High-level descriptions of tests performed in NetApp R&D laboratories designed to demonstrate the benefits of integrating NetApp FAS storage with the ParAccel Analytic Database
- Descriptions of test environments used
- Observed test results
- Interpretation of test results in the context of use cases, industry trends, and the challenges enterprise organizations face in management and analysis of data that has grown so large in volume as to render traditional RDBMS systems ineffective

This document provides the additional level of procedural detail required for an interested reader of the ESG report referenced above to do the following with the assistance of a qualified ParAccel field engineer:

- Repeat the backup and recovery and FlexClone-based tests described in the ESG report.
- Leverage the ParAccel PADB and NetApp SAN optimized solution for better management and more effective analytics of "big data."

While not a requirement, we do recommend that you read the ESG test report mentioned above because it provides additional context around the features described in this paper.

1.2 SCOPE

This report provides additional detail for deploying the ParAccel Analytic Database on NetApp storage in a SAN environment utilizing PADB's Blended Scan technology, as described in ESG's lab validation report ParAccel PADB and NetApp SAN Optimized Solution.

The best practices and recommendations set forth in this paper enable a highly available, easy-to manage environment that provides storage resiliency, scalability, and recoverability for the ParAccel PADB. The target audience is expected to have a basic understanding of databases in general, NetApp architecture, and ParAccel's PADB.

More specifically, we address each topic discussed in the previously referenced ESG report in sufficient detail to enable you to repeat the validation tests and begin to leverage the validated features for your organization.

Procedures described in this document are limited to the test scenarios described in the ESG validation report. Topics to be covered include data protection and virtual scaling of data analytics using NetApp FlexClone. PADB is a complex system that should only be set up with the assistance of a qualified ParAccel field or support engineer. As a result, we do not discuss the actual procedure of setting up a PADB cluster or the Blended Scan configuration.

Before diving into procedures and results, we need to review some PADB basics and NetApp features, which should heighten your understanding of the information that follows.

2 OVERVIEW

The ParAccel Analytic Database is the fastest, most efficient choice for analytic data warehousing workloads, from analytic reporting to ad hoc analysis to highly complex predictive/prescriptive use. Using PADB, you derive deeper insights faster, enabling you to establish an ongoing, open dialog with your data. NetApp provides innovative storage and data management solutions that deliver outstanding cost efficiency and accelerate performance breakthroughs. Together, NetApp and ParAccel provide a superior data warehousing and analytics platform, delivering both the performance and protection required for your largest and most business-critical data analyses. Our combined modular approach enables you to take advantage of the latest improvements in storage hardware and software to maximize results and minimize operational costs in time frames that matter.

Business analysts want easy access to vast amounts of data, high performance on even the most complex of analytic queries, and flexibility. IT wants the ability to deploy and manage solutions as seamlessly as possible, at a reasonable cost.

ParAccel's patent-pending Blended Scan technology satisfies both these needs by dynamically balancing I/O across direct-attached and network-attached storage. This approach delivers enhanced performance on analytic workloads via increased I/O throughput and CPU utilization, while leveraging the data management and data protection capabilities of your SAN or NAS.

The ParAccel Analytic Database is a massively parallel database with a shared-nothing architecture that provides near-linear scalability. PADB plans and executes even the most complex analytic queries in record time because of its unique architecture:

- PADB delivers near-linear performance and data volume scaling by utilizing a Massively Parallel Processing (MPP) architecture that executes database workloads in parallel on multiple commodity hardware–based compute nodes.
- PADB's Massively Parallel Processing, columnar-aware, patent-pending Omne Optimizer creates efficient cost- and rules-based query plans, enabling you to leverage advanced analytic features like window aggregates, affinity, complex table joins, and correlated subqueries.
- PADB's columnar storage architecture only reads relevant data for the query and enables high compression ratios, which in turn provides industry-leading I/O throughput and efficiency.
- PADB's Blended Scan feature enables it to be "storage aware" by managing the direct-attached storage and SAN in such a way as to maximize scan speeds.
- PADB is the only next-generation analytics platform that compiles queries to maximize CPU throughput on modern-day multicore processors.
- PADB uses a high-speed interconnect that delivers ultrafast internode communication.

To support the PADB features listed above, NetApp provides the following:

- NetApp Snapshot solutions for fast backup and restore in the event of data loss or user-generated corruption—these Snapshot copies can be created during ETL operations, allowing fast recovery and thus avoiding ETL failures.
- NetApp FlexClone capabilities provide fast and storage-efficient PADB clones used to generate fully functional copies for creating data marts, dev/test and QA environments, as well as additional reporting environments.
- NetApp unified storage provides a modular design that enables you to scale capacity and performance as needed.
- NetApp RAID-DP® maintains PADB data availability even in the event of a double disk failure. That, combined with NetApp's implementation of hot spare disks, provides 24/7 availability.
- NetApp storage controller failover provides data availability in the unlikely event that a FAS controller fails, also enabling 24/7 availability.

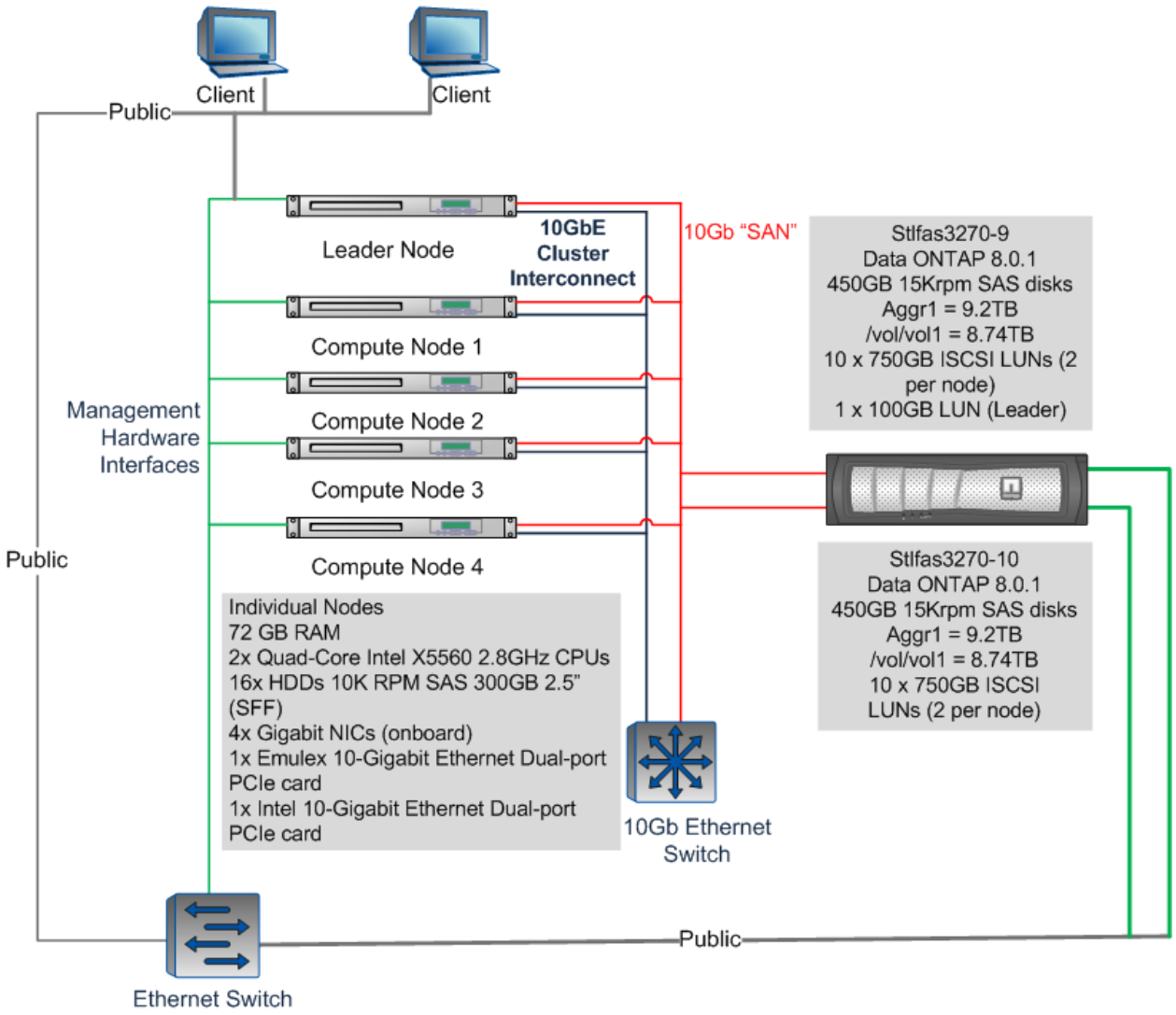
2.1 ARCHITECTURE

Figure 1 shows the environment that was used to perform the backup and recovery tests. It consists of a cluster of five nodes running the PADB database, each with internal storage supplemented by NetApp storage and using the Blended Scan approach, in this case a FAS3270. There are, in this case, four networks with the following functions:

- Dedicated 10GbE cluster interconnect for PADB internal node communication
- Hardware management network
- Public network for client access
- iSCSI-based storage area network (SAN) to provide the external storage required for enablement of ParAccel's Blended Scan technology

Specific hardware and software details are noted in the diagram and discussed in the following sections.

Figure 1) Solution overview.



2.1.1 SOLUTION COMPONENTS—PARACCEL

At the highest level, a PADB system has four main architectural components:

- Leader node ("leader")
- Compute nodes
- Parallel communication fabric
- An optional storage area network (SAN)

The leader node controls the execution of the compute nodes, and all nodes communicate with each other via the dedicated fabric. Leader and compute nodes are standard x86 servers running Linux®.

Users and applications communicate with the system via the leader node by using standard interfaces—ANSI SQL via ODBC/JDBC.

LEADER NODE

The leader sits on the customer's network and is the only PADB node intended to interface with external applications and the rest of the IT infrastructure. The leader communicates with applications and users via standard ODBC or JDBC, and recognizes ANSI SQL plus PADB extensions.

A leader is required to manage communication with the compute nodes. The leader is responsible for controlling sessions, scheduling execution, parsing, optimizing, and producing object code executing queries. In our configuration, the PADB leader node was not configured as a "shared leader" and, as a result, did not participate in executing database queries for this scenario.

Architectural workload separation by node type (leader and compute) allows better throughput optimization—the leader's bandwidth is optimized for outward communication and handling of query overhead so each compute node's bandwidth is dedicated to data operations.

COMPUTE NODE

Compute nodes are the machines responsible for processing and storing data. Data can be stored directly on compute nodes all in memory, on SSD as well as on DASD, as well as on a SAN, enabling the use of PADB's patent-pending Blended Scan technology. Each node stores and manages a subset of the rows of each table. For example, if a table with 10 columns has 1 billion rows and there are 20 compute nodes, then the data for about 50 million rows are distributed to each node.

Data is distributed to a particular node based on a hashing algorithm applied to a distribution key, or by round robin. Distribution keys, such as the primary key or other popular joins, are good for even distribution of data, especially when queries will benefit from collocated joins by using the same distribution key. In cases in which an inherently balanced distribution key isn't obvious or doesn't exist, round-robin distribution can be used to balance the data. By offering multiple methods of data distribution, it is possible to maintain the appropriate balance between data distribution and performance so PADB can take best advantage of its resources and provide good parallel efficiency.

PADB performance is driven by how many compute nodes are present. For example, with most applications, a 50-compute node system will perform 5 times faster than a 10-compute node system. Therefore, performance and price/performance are inextricably linked on a PADB system.

COMMUNICATION FABRIC

The ParAccel Communication Fabric is a low-cost, high-performance fabric based on standard, ubiquitous Gigabit Ethernet (GbE) and standard multiport switches that have full crossbar support (a suggested model is the Cisco 3750). It uses a custom ParAccel Interconnect Protocol to enable highly efficient communication among all of the nodes (leader and compute). It delivers maximum interconnect performance because it is specifically designed for how traffic moves in a complex, parallel database environment (for example, large intermediate result sets, data redistribution), and therefore uses multiple links simultaneously running multiple data streams. The fabric is implemented internally as multiple independent networks all working on behalf of the database. Although at least two GbE fabrics are required for high availability, PADB will utilize as many fabrics as are available for increased performance. The fabric is also fully compatible with 10GbE, which is growing in popularity due to its lower latency and higher throughput capabilities.

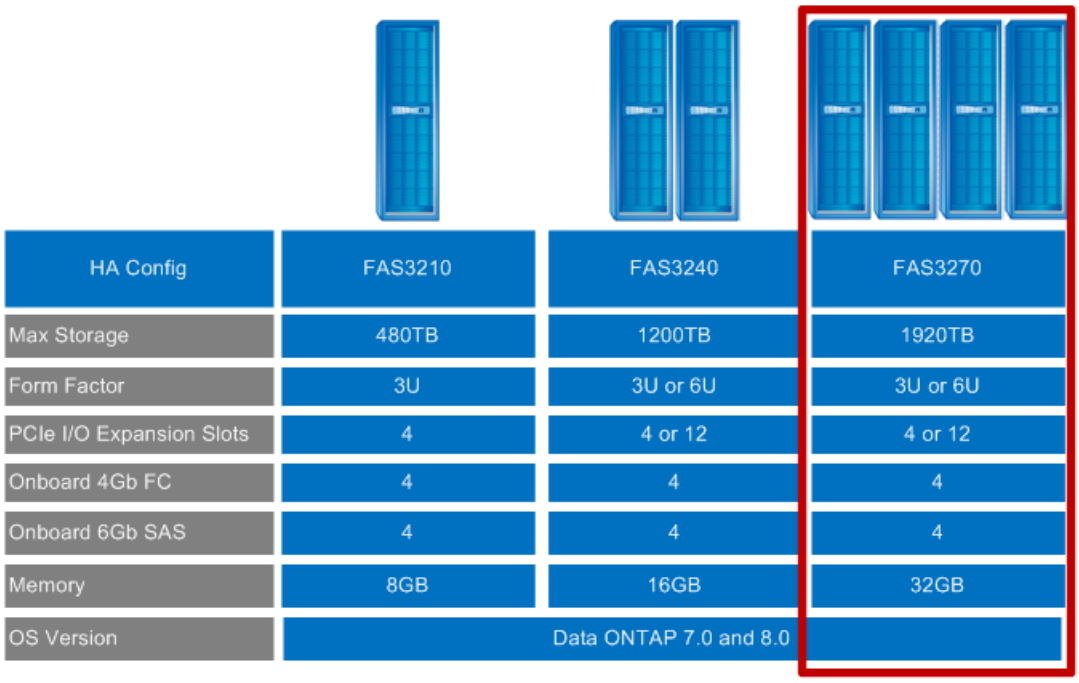
2.1.2 SOLUTION COMPONENTS—NETAPP STORAGE SYSTEMS

For these tests, we used the FAS3270 unified storage platform. The FAS3200 series storage systems are great choices for use as building blocks for a customer's storage architecture. The FAS3200 series offers the capability to add more storage while protecting your investment in other storage platforms. The FAS3200 series seamlessly integrates into multistorage platform environments and offers a compelling value proposition that will benefit business by:

- Providing more storage at a lower cost
- Increasing data availability to keep a customer's business operating efficiently
- Managing enterprise-wide storage requirements with a single operating system

Figure 2 depicts the three storage platforms available to provide flexibility, scalability, and performance to meet your future business needs.

Figure 2) FAS system summary.



SNAPSHOT COPY AND RESTORE

Snapshot technology is available from a variety of data storage vendors, but not all snapshot technologies are created equal. NetApp Snapshot enables IT administrators to create space-efficient, point-in-time copies of virtual machines or entire datastores that do not impact overall performance or consume any additional storage until the blocks held in the snapshot are overwritten. Then, using SnapRestore, you can restore from these backup copies at any level of granularity—single files, directories, or entire volumes—simply and quickly when required. Restores can be done rapidly from any of the Snapshot copies, providing customers with an exceptional recovery time objective (RTO).

Up to 255 Snapshot copies can be created automatically or manually on each volume. Customers can use NetApp Snapshot technology to perform backups as often as needed—daily, hourly, and so on. In the event of a recovery, more frequent backups will reduce data lost since the last backup was taken. This greatly enhances a customer's recovery point objective (RPO). Each Snapshot copy is RAID protected for reliable backup.

FLEXCLONE

With NetApp FlexClone technology, administrators can create instant writable Snapshot copies to support multiple uses including testing, reporting, and development.

Unlike full copies from mirrored production data, FlexClone copies enjoy the same benefits as Snapshot copies described above while presenting a writable copy of your data that consumes very little additional space and does not impact your overall performance.

With FlexClone, you can affordably create several clones without having to add more storage.

3 IMPLEMENTATION

3.1 PARACCEL ANALYTIC DATABASE

SYSTEM

There is a wide variation in how the hardware for PADB clusters can be configured for any given project, involving different SAN, disk, and internal PADB network configurations. With the PADB Blended Scan technology, the size and speed ratio between the SAN and local disk determine the ultimate database size. We worked closely with ParAccel to correctly size the configuration to be used in our tests.

PADB installation is a guided process that is initiated by booting the leader node from a PADB installation CD-ROM. That CD-ROM installs the latest supported release of the CentOS operating system and PADB on the leader(s) and compute nodes. Extra setup and license keys may be required to configure leader node failover using Veritas™, along with any required NetApp software.

Best Practice

Configure the PADB hardware and perform the installation before connecting a SAN to the PADB cluster. After the SAN is configured, the relative SAN speeds can be set in the PADB configuration file, or calculated by PADB using the PADB `ANALYZE DISK SPEEDS` command. The PADB Blended Scan technology is integral to the database and does not require any unique setup past storing PADB data on the SAN.

For NetApp Snapshot and FlexClone technology to work, the PADB leader node must be configured with the PADB installation directory located on a NFS mount pointing to `/home/paraccel/padb`. The PADB installation directory holds metadata and some database operational data that must be in sync with any Snapshot copy of the data volumes used on the PADB compute nodes.

The number of nodes does not matter; however, for the sake of example, the tested PADB configuration had four active compute nodes and one leader node. The PADB `/home/paraccel/padb` installation directory on the leader node was created on a volume on storage controller 1 named `/vol/ vol3`, mounted as `/mnt/paraccel`, and symbolically linked to `/home/paraccel/padb`. We also created a pair of 750GB LUNs for each compute node on the NetApp FAS3270 to be used for Blended Scan functionality. The LUNs in each pair were created in different volumes on different controllers in order to evenly spread the workload across the two controllers of our FAS3270 systems. For details of storage sizing and provisioning, see Table 1.

The LUNs appear as raw devices to PADB, and are managed by udev to provide a consistent name and ordering of the devices. If the udev names are `lun_data*`, then the `san_list` setting in the `xenpostgresql.conf` file would look like:

`disk_list|raw,cciss/c0d0p1,cciss/c0d1p1…`

`san_list|lun_data0,lun_data1…`

The `xenpostgresql.conf` file is the PADB configuration file. It resides on all nodes in the cluster. Modifications must be made on the leader node, after which it must be copied to all compute nodes using a PADB utility. In this example, we have defined raw device names for the internal disk drives of the compute nodes using the `disk_list` parameter and the udev names corresponding to our NetApp LUNs using the `san_list` parameter.

3.2 NETAPP UNIFIED STORAGE

SYSTEM

A FAS3270 dual-controller system was employed. Each controller was configured with:

- Data ONTAP 8.0.1
- 1 1/2 x DS4243 storage shelves (3 shelves shared between 2 controllers)
  36 x 450GB 15Krpm SAS disks per controller
- 10Gb Ethernet connectivity

LICENSES

Licenses installed on each controller included:

- iSCSI
- Cluster
- FlexClone
- SnapRestore

AGGREGATES/VOLUMES/LUNS

The storage layout on the FAS3270 was as follows.

Table 1) Storage provisioning details.

Controller	Aggregate/size	Volume/Size	LUN/Size	Purpose
STLFAS3270-9	Aggr0	Vol0	-	Root
	Aggr1/9.2TB	Vol1/8.74TB	LUN0,1/750GB each	Leader Node
			LUN2,3/750GB each	Compute Node 1
			LUN4,5/750GB each	Compute Node 2
			LUN6,7/750GB each	Compute Node 3
			LUN8,9/750GB each	Compute Node 4
			LUN10/100GB	Leader Node
		Vol3/4TB		Leader Node installation directory
STLFAS3270-10	Aggr0	Vol0	-	Root
	Aggr1/9.2TB	Vol1/8.74TB	LUN0,1/750GB each	Leader Node
			LUN2,3/750GB each	Compute Node 1
			LUN4,5/750GB each	Compute Node 2
			LUN6,7/750GB each	Compute Node 3
			LUN8,9/750GB each	Compute Node 4

Please note in Table 1 above that each compute node uses two LUNs from each controller to support ParAccel's Blended Scan configuration. The nodes were split across the two controllers (four LUNs total for each compute node) in order to balance the workload across the two FAS systems. This configuration improves the performance of the Blended Scan configuration, performance during redistribution of data to compute nodes, and the speed of the actual redistribution operation, eliminating performance bottlenecks due to uneven distribution of I/O loads across the storage controllers.

4 TESTS

4.1 DATA PROTECTION

The purpose of this test, as executed for ESG validation, was to showcase the data protection capabilities of the NetApp-ParAccel architecture. To accomplish this we created a scenario in which a data load fails due to end-user error, resulting in a table mistakenly created and loaded. To recover from the error, we restore the initial PADB instance from a NetApp Snapshot copy created before the error. A detailed outline of our procedure follows.

4.1.1 DATA PROTECTION—DETAILED PROCEDURE

1. Create a test database. Perform a timed SQL® query against the database. For this test we reused a database created during an internal NetApp POC, consisting of sanitized NetApp AutoSupport™ (ASUP™) data. That database initially consisted of 85 tables with no indexes and a total of 6.22 billion rows, and was 2.6TB in size. (This database represents our data load before the error.)
2. Perform a timed row count of every table in our database, and record the elapsed time and number of rows.
3. Create NetApp Snapshot copies of the database volumes.
4. Load incremental data into the database (that is, "mistakenly" create and load a new table), thus creating our error condition requiring a rewind back to the pre-error state of the database.
5. Observe that the post-Snapshot table and data exist by performing the same query as above and record the results (elapsed time and row count). The row count should be higher than before creating the Snapshot copy.
6. Stop the ParAccel Analytic Database.
7. Use SnapRestore to go back to the previous Snapshot copy.
8. Bring the PADB online.
9. Observe using the same SQL statements above (timed row count) that the table and data created after the Snapshot copy are not present. The row count should be the same as before the Snapshot copy was created.
10. Rerun the SQL query performed at the beginning of this test and record the elapsed time twice, once while data is still being redistributed from the SAN to the compute node local disks and once after redistribution has completed. Record both elapsed times.
11. Observe the performance impact of background repopulating of data to local drives on the PADB compute nodes.

As mentioned earlier, our ParAccel Database is configured with four active compute nodes and one leader node and the database files were spread across multiple FlexVol® volumes on both FAS3270 storage controllers. Because there was more than one FlexVol volume involved, we used NetApp's consistency group API to create Snapshot copies. With NetApp consistency group Snapshot technology, there is no need to shut the database down or to limit access while creating Snapshot copies. The procedure used in our testing is outlined below, where the **cgsnap** command is a compiled Perl script provided by NetApp (see Appendix 6.2).

4.1.2 STEP-BY-STEP PROCEDURE

The steps we followed for ESG testing are listed below:

1. Create a test database. Our test setup uses a preexisting database that was created for an internal NetApp POC.
2. Run a timed set of SQL queries to perform a row count of all tables in the database. This establishes a baseline for future queries.
3. Create Snapshot copy. Note that the database does not need to be shut down.
4. From a leader node you might run the following sample commands to clone the volumes:

   ```
   export SNAPNAME=snap_PADB
   export CGSNAP_VERBOSE=1
   cgsnap $SNAPNAME stlfas3270-9:vol1 stlfas3270-9:vol2 stlfas3270-10:vol1
   ```

5. Load incremental data into the database (that is, "mistakenly" create and load a new table), thus creating our error condition requiring a rewind back to the pre-error state of the database.
6. Observe that the post-Snapshot copy table and data exist by performing the same query as above and record the results (elapsed time and row count). The row count should be higher than before creating the Snapshot copy.
7. Correct the error condition by restoring the Snapshot copy on the PADB leader node as follows:
   a. Stop PADB:
      ```
      cqi xstop
      ```
   b. Stop iSCSI:
      ```
      cqi allx ‘sudo service iscsi stop’ 2>&1 > /dev/null
      ```
   c. Stop PADB System Manager:
      ```
      cqi stop sysmgr
      ```
   d. Unmount PADB installation directory from NFS:
      ```
      sudo umount /mnt/paraccel
      ```
   e. Restore the Snapshot copy:
      ```
      ssh root@stlfas3270-9 snap restore -f -t vol -s $SNAPNAME vol1
      ssh root@stlfas3270-9 snap restore -f -t vol -s $SNAPNAME vol2
      ssh root@stlfas3270-10 snap restore -f -t vol -s $SNAPNAME vol1
      ```
   f. Mount PADB installation directory:
      ```
      sudo mount /mnt/paraccel
      ```
   g. Remove old postmaster pid:
      ```
      rm /home/paraccel/padb/data/pg/postmaster.pid
      ```
   h. Distribute PADB binaries to compute nodes:
      ```
      cqi distbin
      ```
   i. Start PADB System Manager:
      ```
      cqi start sysmgr
      ```
   j. Start iSCSI on all nodes:
      ```
      cqi allx  'sudo service iscsi start'
      ```
   k. Set PADB to restore data to local disk from SAN on startup:
      ```
      cqi restore from san
      ```
   l. Start PADB:
      ```
      cqi xstart
      ```

m. Run the same SQL query as the one used before snapshot creation and record the elapsed time. This step is to be done during redistribution of data from SAN to disks and after redistribution has completed.

Note that the redistribution of data from our SAN (FAS storage) to the local disks on the compute nodes began immediately after PADB startup described in step l, above. After starting the database, the data on the compute nodes had to be reloaded from the copy of record on the SAN. During this redistribution process, the performance of the database was not optimal and did not return to preload levels until the process was complete and the Blended Scan balance between the compute nodes and the SAN was restored. Once complete, we found that the performance returned to pre-error levels. To measure the performance, we ran timed SQL queries before using SnapRestore, during redistribution, and after redistribution. Below are the results.

1. Initial query timing and row count (with Blended Scan, before creating the Snapshot copy): 14.178 seconds and 7,854,048,951 rows
2. Query timing and row count (with Blended Scan, after creating the Snapshot copy): 14.178 seconds and 7,854,048,951 rows, demonstrating that the existence of the Snapshot copy did not impact the performance of the database
3. Row count after failed data load: 7,854,048,955 rows
4. Results after restoring from the Snapshot copy, during redistribution to the local compute node disks (no Blended Scan): 3 minutes, 40 seconds, and 7,854,048,951 rows counted
5. Observation after restoring the Snapshot copy and data redistribution has completed (with Blended Scan): 15.179 seconds and 7,854,048,951 rows
6. Data redistribution required 6 minutes and 42 seconds to complete

The results are summarized in Table 2, below.

Table 2) Row count and timing results during the data protection test.

PADB Status	Row Count	Elapsed Time (seconds)
Blended Scan, Before Snapshot	7,854,048,951	14.178
Blended Scan, After Snapshot	7,854,048,951	14.178
Blended Scan, After Failed Data Load	7,854,048,955	N/A
After SnapRestore, During Data Redistribution	7,854,048,951	220.000
After SnapRestore, After Data Redistribution (w/Blended Scan)	7,854,048,951	15.179
Data Redistribution	N/A	402.0

As you can see, restoring the Snapshot copy of the PADB volumes successfully rolled the database back to its error-free state with no impact on query completion time once the data redistribution process had completed. In this way, we were able to avoid the necessity of a complete reload of the database. Also, remember that the Snapshot copy used for recovery was created almost instantaneously with the database up and running. Restoring the Snapshot copy required only a few seconds to complete with the data being completely available immediately afterward.

4.2 VIRTUAL SCALING WITH NETAPP FLEXCLONE TECHNOLOGY

The purpose of this test, as executed for ESG validation, was to demonstrate the unique capability of a combined NetApp and ParAccel solution to scale data access through NetApp storage virtualization (that is, FlexClone) technology. Database clones created in this manner require very little incremental storage and can be used for creation of "instant data marts" for QA, for additional reporting, and to provide test/development environments. Below is a basic outline of our procedure.

1. Create a 3-node PADB cluster and create a small TPC-H database on it. Our database consisted of 8 tables without any indexes or tuning objects, 866 million rows, 42GB in size.
2. Run a query benchmark and record timings as a baseline. For our tests, we used the following SQL statement:

   ```
   select count (distinct c_address), count(distinct c_custkey) from customer;
   ```
3. Configure another three-node PADB cluster.
4. Using FlexClone, clone the database from the initial cluster to the new three-node cluster.
5. Start the cloned PADB cluster.
6. Measure the performance of the cloned PADB cluster using the same query described above.

After starting up the second PADB cluster, the database redistribution process starts to populate the three nodes in the new cluster using the database of record on the SAN. In this case, the database on the SAN being used is actually the cloned copy of the original database. There is no impact on the original database used by the first PADB cluster. Once the redistribution is complete, the end result is two identical three-node PADB clusters, each with its own writable copy of the original database that is consuming only 1% more storage compared to the initial three-node PADB cluster. In essence, a fully functional copy of the 42GB PADB database is available for access while requiring only 1% additional storage space.

After completing the redistribution process on the second PADB cluster, we ran the same query defined above on the second PADB cluster and validated that the overall performance was the same on both PADB clusters and that this performance was comparable to the performance observed before the FlexClone volume of the original database was created. In other words, we found that performance was comparable whether running from the original or the cloned database.

4.2.1 CLONING A PADB CLUSTER USING NETAPP FLEXCLONE AND ISCSI—DETAILED PROCEDURE

This section provides the steps required to use the NetApp FlexClone technology to create a writable clone of the PADB database and assign it to a second PADB cluster. For the success of this procedure, the following conditions should be met:

- The source and target PADB clusters must have the same topography. The number of nodes, disks, and cores need to be consistent between the clone source and the target clusters.
- PADB has been installed on the clone target and configured with a working leader and compute nodes.
- NetApp FlexClone volumes are created from volume Snapshot copies. Consistency group functionality must be used in creating those Snapshot copies if the source volumes span multiple NetApp storage controllers, and if the Snapshot copies are to be created from an active database (that is, the database is up, running, and accessible to users).
- LUNs used by the compute nodes must have consistent device names across all of the compute nodes within a PADB cluster. For example, if device names `/dev/lun_data0` and `/dev/lun_data1` are used on one compute node, those names must be used for the respective LUNs on all the

compute nodes. This naming convention must be persistent across reboots and must be accomplished using udev rules.

- Cloned LUNs must be assigned the same persistent device names as the corresponding source LUNs on the source cluster. For example, you must use the same mapped device name for the clone of LUN `/vol/vol1/c1l4` as for the LUN `/vol/vol1/c1l4` itself, so you might map the device name /dev/lun_data0 to both of them using udev rules. The sanlun utility of NetApp's iSCSI support tools must be used to obtain the storage-side names for the LUNs. Below is sample output from running the **`sanlun lun show`** command on the compute node:

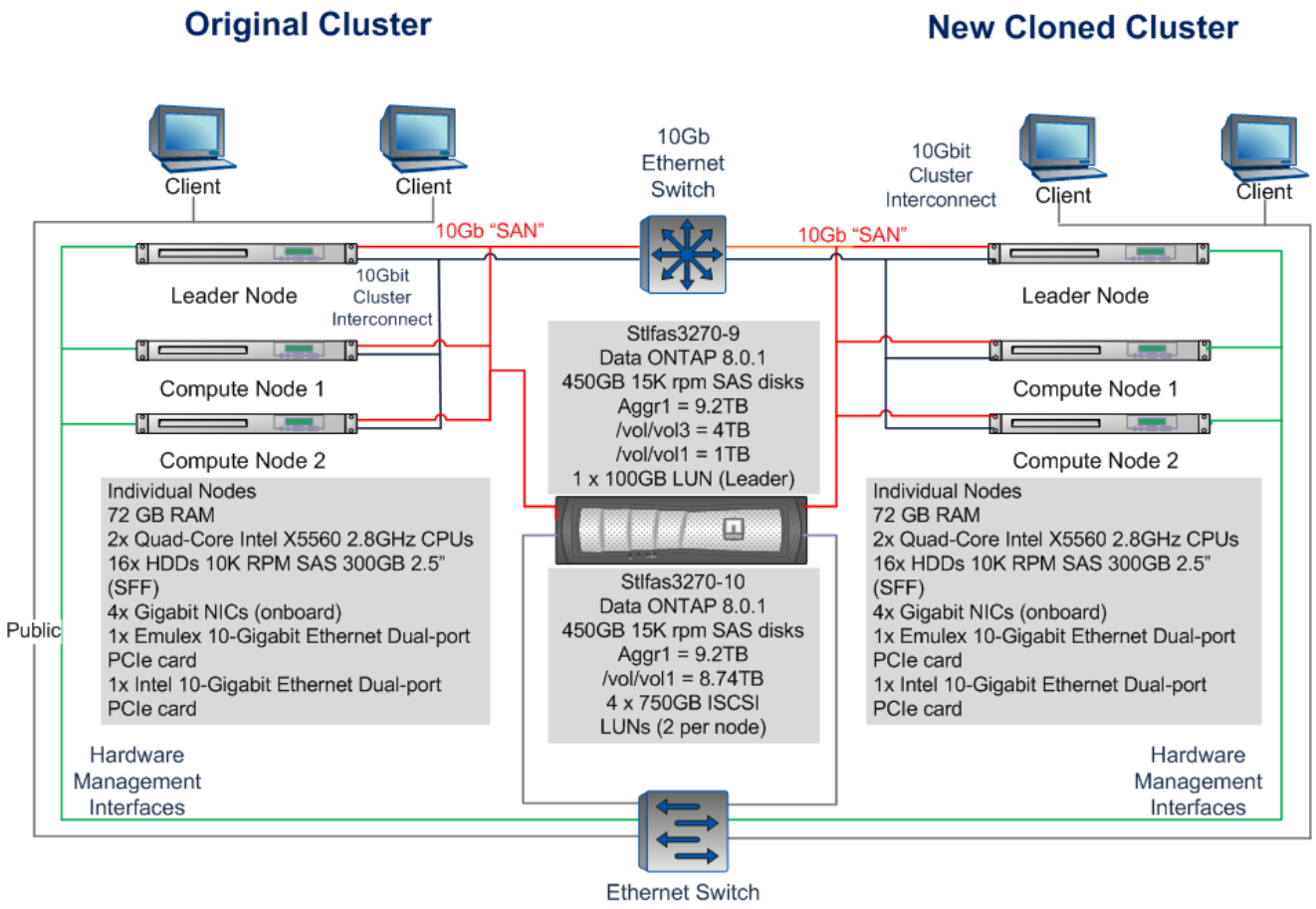
```
controller:    lun-pathname     device    adapter    protocol  lun size
stlfas3270-10: /vol/vol1/c1l4   /dev/sdc  host23     iSCSI      750.1g (805371379712)
stlfas3270-10: /vol/vol1/c1l3   /dev/sdb  host23     iSCSI      750.1g (805371379712)
```

7. NetApp's consistency group API is required for creating consistent Snapshot copies across storage controllers.
8. NetApp's iSCSI host utilities must be installed on all compute nodes. These utilities can be downloaded from the NetApp Support site.

4.2.2 STEP-BY-STEP PROCEDURE

The steps we followed for ESG validation are listed below. Due to hardware limitations, the FlexClone use case required reconfiguration of our original six-node cluster and resulted in a three-node source cluster and a three-node cloned cluster, as illustrated in Figure 3, below.

Figure 3) Cloned PADB environment.



Note the original cluster on the left and the cloned cluster on the right. They each consist of one leader node and two compute nodes, and use the same FAS3270 storage system. On a per-node basis, storage was provisioned on our clone source in the same way as with our backup and recovery test, with each compute node having two LUNs. Unlike the backup and recovery test, we used a TPC-H database with a "select count" query against the customer table for our FlexClone test.

Below are the steps followed to create the PADB volume/LUN clones for the cloned cluster and to make them available to the cloned cluster nodes. These commands were run from a script on the leader node.

1. Create Snapshot copies of the volume used for the PADB installation directory (on the leader node) and the volume containing data LUNs using the NetApp consistency group API.

   ```
   export SNAPNAME=clone_snap_PADB
   export CGSNAP_VERBOSE=1
   ./cgsnap $SNAPNAME stlfas3270-9:vol3 stlfas3270-10:vol1
   ```

2. Create clones of the PADB installation volume and the data LUN volume using the previously created Snapshot copies.

   ```
   ssh root@stlfas3270-9 vol clone create vol3_clone -s none -b vol3 $SNAPNAME
   ssh root@stlfas3270-10 vol clone create vol1_clone -s none -b vol1 $SNAPNAME
   ```

3. Bring the LUNs in the cloned volumes online.

   ```
   export SNAPNAME=clone_snap_PADB
   ssh root@stlfas3270-10 lun online /vol/vol1_clone/c1l3
   ssh root@stlfas3270-10 lun online /vol/vol1_clone/c1l4
   ssh root@stlfas3270-10 lun online /vol/vol1_clone/c2l3
   ssh root@stlfas3270-10 lun online /vol/vol1_clone/c2l4
   ```

4. Create an iSCSI initiator group for LUNs to be used by each cloned compute node. We need one initiator group for each cloned compute node.

   ```
   ssh root@stlfas3270-10 igroup create -i -t linux c1_clone
   ssh root@stlfas3270-10 igroup create -i -t linux c2_clone
   ```

5. Map the cloned LUNs to the appropriate initiator group, to give each cloned compute node access to two LUN clones.

   ```
   ssh root@stlfas3270-10 lun map /vol/vol1_clone/c1l3 c1_clone 3
   ssh root@stlfas3270-10 lun map /vol/vol1_clone/c1l4 c1_clone 4
   ssh root@stlfas3270-10 lun map /vol/vol1_clone/c2l3 c2_clone 3
   ssh root@stlfas3270-10 lun map /vol/vol1_clone/c2l4 c2_clone 4
   ```

6. Add the iSCSI initiator identifier of each cloned compute node to the initiator group it will use for access to its cloned LUNs.

   ```
   ssh root@stlfas3270-10 igroup add c1_clone iqn.1994-05.com.redhat:8a985ae4a01c
   ssh root@stlfas3270-10 igroup add c2_clone iqn.1994-05.com.redhat:3bda8f46d10
   ```

7. Point the PADB compute node SAN disks to the cloned LUNs on the second target cloned system.

   a. Mount the cloned PADB directory:

      ```
      sudo mount /mnt/paraccel
      ```

   b. Start the PADB System Manager:

      ```
      cqi start symgr
      ```

   c. Start iSCSI on the compute nodes:

      ```
      cqi allx -qq -a COMPUTE 'sudo service iscsi restart'
      ```

   d. Get persistent device names on all PADB compute nodes:

      ```
      cqi allx -qq -a COMPUTE 'sudo iscsiadm -m discovery -t sendtargets -p
      192.168.1.9'
      ```

e. On each compute node, configure udev rules to point to cloned devices:

`sanlun lun show` **to show LUN names and /dev/sd* mapping**

`scsi_id -gxs /block/sda`

`scsi_id -gxs /block/sdb`

`scsi_id -gxs /block/sdc`

and so forth

Edit the **`/etc/udev/rules.d/99-static-iscsi-names.rules`** file and change iSCSI LUN serial numbers to match those obtained in the previous step.

See Appendix 6.2, "UDEV Rules—Example," for additional details and a sample "99-static-iscsi-names.rules" file.

Then run the following commands from the leader node to reload the udev rules and start udev:

`sudo udevcontrol reload_rules`

`sudo start_udev`

f. Set PADB to restore data to the local disk from SAN on startup:

`cqi restore from san`

g. Start PADB:

`cqi xstart`

RESULTS AND COMMENT

Using scripts, creation of our "Instant Data Mart" required 1 minute and 8 seconds. After data redistribution completed to the "cloned" PADB cluster, performance with the clone was comparable to that of the original database. We observed the following completion times for our test query:

Query of the source database only: 28.654 seconds

Query of the database clone only: 35.355 seconds

Query of both source and clone simultaneously:

Source database: 28.703 seconds (almost the same as query of source only)

Database clone: 32.311 seconds (very close to source query completion time)

These observations are summarized in Table 3, below.

Table 3) Query completion times for source PADB and instant data mart (PADB clone).

Query Description	Source PADB Query Elapsed Time (seconds)	PADB Clone Query Elapsed Time (seconds)
Query of Source Database Only	28.654	N/A
Query of Database Clone Only	N/A	35.355
Query of Both Source and Clone Simultaneously	28.703	32.311

Although this test was performed on a very small scale, the same process can be applied successfully to any size database and/or cluster node count.

5 CONCLUSION

The ParAccel Analytic Database with NetApp storage provides an excellent platform for large-scale business-critical data warehouse applications. Using ParAccel's patent-pending Blended Scan technology, complex queries can be performed against extremely large datasets efficiently, quickly, and economically. With Blended Scan technology, the PADB/NetApp solution enables enterprise organizations to fully realize the competitive advantages of knowledge previously locked within vast amounts of data—datasets so large that traditional RDBMS systems were completely ineffective. ParAccel databases are also highly scalable. Additional key features of this solution include the following:

- Extremely fast, storage-efficient backup and restore capabilities using NetApp Snapshot technology. NetApp Snapshot copies require very little storage space, are created almost instantaneously, and can be restored in a matter of seconds. In the Blended Scan configuration, data becomes available to the user immediately after using SnapRestore.
- Fast creation of storage-efficient PADB clones using NetApp FlexClone technology. PADB clusters can be cloned quickly, requiring very little storage space upon creation. They can be used to provide virtual environments for QA, additional reporting, test/development environments, and more.

The NetApp-ParAccel solution is powerful, fast, reliable, and scalable. It provides all the features required to meet the challenges of today's enterprise organizations in management, protection, and full utilization of the vast wealth of data available from within and outside the enterprise.

This document stands on its own as a guide for implementing NetApp's data protection and cloning technologies with PADB clusters; however, we strongly recommend that you also read the referenced ESG lab validation report, "ParAccel PADB and NetApp SAN Optimized Solution": http://www.enterprisestrategygroup.com/media/wordpress/2011/05/ESG-Lab-Validation-Report-NetApp-ParAccel-May-11.pdf.

6 APPENDIXES

6.1 COMMANDS

CQI COMMANDS

Command	Description
cqi distbin	Redistributes ParAccel binaries to compute nodes
cqi allx uptime	Runs Linux `uptime` command on all nodes and returns
cqi stop|start sysmgr	Stop/Starts PADB System Manager
cqi xstop/xstart	Stop/Starts PADB
cqi initdb	Reinitializes PADB (overwrites current database)

POSTGRESQL/PARACCEL COMMANDS

Command	Description
psql dev	Logs into PADB "dev" database after the database has been started
\q	Quits the database
\d table_name	Describes table
\dS	Lists system tables and views

\i filename	Executes a program or script; in this example we are executing a script named "filename"
\! host_command	Executes host command (that is, `\! ls -ltr`) from within PADB
select version();	Retrieves PADB version information in PSQL

analyze disk speeds;	Runs test to get disk speeds (local and SAN), and determines the relative speed ratio used for Blended Scan; should be run once at setup time; use query below to retrieve information:
	`Select host, diskno,`
	`trim(mount) as disk, mbps`
	`from stv_partitions`
	`order by host, diskno;`

6.2 UDEV RULES—EXAMPLE

In Section 4.2.2 above we define a step-by-step procedure for cloning a PADB cluster using NetApp FlexClone. In step 5 of that section, we provided a basic procedure for configuring udev rules to provide consistent iSCSI device names for both the original PADB cluster and the clone, and to provide device name persistency across reboots. As previously stated, new udev rules must be added to the `/etc/udev/rules.d/99-static-iscsi-names.rules` file for implementation. Here is a sample `99-static-iscsi-names.rules` file containing three new iSCSI rules:

```
KERNEL=="sd*1", BUS=="scsi", PROGRAM=="/sbin/scsi_id",
RESULT=="360a98000646651355434626252 4c7533", NAME="lun_data0", OWNER="root",
GROUP="paraccel", MODE="0660"

KERNEL=="sd*1", BUS=="scsi", PROGRAM=="/sbin/scsi_id",
RESULT=="360a98000646557664a6f6262524b5978", NAME="lun_data1", OWNER="root",
GROUP="paraccel", MODE="0660"

KERNEL=="sd*1", BUS=="scsi", PROGRAM=="/sbin/scsi_id",
RESULT=="360a98000646557664a6f6262524d656b", NAME="lun_data2", OWNER="root",
GROUP="paraccel", MODE="0660"
```

In this example, each new rule begins with the keyword “KERNEL,” which defines the initial iSCSI device name to be changed. In our first rule, udev renames any device identified as “sd*1,” having a unique SCSI ID of “360a98000646651355434626252 4c7533,” to “lun_data0.” (“sd*1” can refer to any device name starting with the character string “sd” and ending with “1.” Examples include “sda1,” “sdc1,” “sdg1,” and so forth.) Using this rule, the owner of that device is set to “root,” the group is set to “paraccel,” and the permissions mode is set to “0660.” As you recall from section 4.2.2, step 5, the unique SCSI ID is obtained using the Linux command **`scsi_id -gxs /block/sd*`** with the **`sd*`** mappings of the NetApp iSCSI devices being obtained using the NetApp sanlun utility. The remaining two rules in the file work the same way as the first one.

For a deeper understanding of udev functionality, please refer to the Linux “udev” man page. For more information on the NetApp “sanlun” utility, see NetApp’s “iSCSI Host Utilities for Linux” at https://now.netapp.com/NOW/cgi-bin/software/?product=iSCSI+Host+Utilities&platform=Linux.

6.3 SCRIPTS FOR CREATING AND RESTORING SNAPSHOT COPIES

The following scripts were created for testing for POCs by NetApp and ParAccel field engineers. They are outside Product Support, have items specific to the POCs (that is, controller names), and should not be considered as production ready. The scripts are provided as one possible path for using NetApp consistency groups and Snapshot copies with PADB.

The scripts include:

- cgsnap.pl—Creates a snapshot consistency group for NetApp storage system/volumes
- crSnap.sh—Creates a unique Snapshot copy name and calls cgsnap
- resSnap.sh—Restores the latest Snapshot copy of PADB data

CGSNAP.PL

NetApp field engineers supplied a Perl script to create consistency groups. Use the compiled version (cgsnap) in calling scripts.

```
#!/usr/bin/perl

# Version 1.0
# arndt@netapp.com
#
# This script will take a consistency group snapshot across multiple
# filers (or vfilers) and multiple volumes using API calls.
#
# Requirements:
#   1. A user with the privileges in the following example must be
#      configured on the storage system or vfiler.
#        > useradmin role add cg_backup_role -a login-http-admin,\
#          api-system-get-version,api-cg-start,api-cg-commit
#        > useradmin group add cg_backup_group -r cg_backup_role
#        > useradmin user add cg_backup -g cg_backup_group
#   2. The password for "cg_backup" should be set to "cg_backup1".
#
# Usage: cgsnap snapshot_name filer|vfiler:volume [filer|vfiler:volume ...]
#
# Example:
#   cgsnap cgsnap.0 filer1:vol1 filer1:vol2 vfiler2:vol1 vfiler3:volZ
#
# Notes:
#   1. To see verbose messages during the execution of this script, set a
#      shell environment variable as such:
#        # export CGSNAP_VERBOSE=1
#   2. This program will return an exit code of 0 for success, and an
#      exit code of 1 for any failure.
#
# (c) 2008 NetApp Inc., All Rights Reserved
#
# NetApp disclaims all warranties, excepting NetApp shall provide support
# of unmodified software pursuant to a valid, separate, purchased support
# agreement.  No distribution or modification of this software is permitted
# by NetApp, except under separate written agreement, which may be withheld
# at NetApp's sole discretion.
#
# THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND ANY EXPRESS OR IMPLIED
# WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
# MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN
# NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
# SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED
# TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
```

```
# PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
# LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
# NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
# SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
#
# Revision History
# 1.0 - Initial release.

use strict;
use warnings;
use NaServer;
use NaElement;

# Global variables.
my ($filer,$volume,$snapshot,%volumes,%cgid,$storageuser,$storagepass);
my ($version,%zapiServer,$verbose);
$version = "1.0";

# See if we are in verbose mode or not.
$verbose = 0;
$verbose++ if ($ENV{'CGSNAP_VERBOSE'});

# Setup storage username and password to use for API connection.
$storageuser = "cg_backup";
$storagepass = "cg_backup1";

# Parse command line.
help("Invalid invocation.") unless ($#ARGV > 1);
$snapshot = $ARGV[0];
for (1..$#ARGV) {
      my $filervol = $ARGV[$_];
      ($filer,$volume) = split /:/, $filervol;
      $volumes{$filer}{$volume}++;
}

# Connect our API handler for each filer in the operation.
for $filer (keys %volumes) {
      connect_zapi($filer);
}

# Start the CG snapshot for each volume in the operation.
for $filer (keys %volumes) {
      cgstart($filer);
}

# Finish the CG snapshot for each volume in the operation.
for $filer (keys %volumes) {
      cgcommit($filer);
}

# Exit cleanly.
exit 0;

# Subroutine to log output messages.
sub logmsg {
      my ($msg,$fatal) = (@_);
      my ($ts);

      unless ($fatal) {
            return unless $verbose;
      }

      $ts = localtime;
```

```
	print STDERR "$ts: $msg\n";

	return 1;
}

# Subroutine to make sure we can communicate with the storagehost.
sub connect_zapi {
	my ($filer) = (@_);
	my ($results);

	# Make sure the login works, or exit if it fails.
	logmsg("Checking connectivity for $storageuser\@$filer.",0);
	$zapiServer{$filer} = NaServer->new($filer,1,1);
	$zapiServer{$filer}->set_admin_user($storageuser,$storagepass);
	$results = $zapiServer{$filer}->invoke("system-get-version");
	if ( $results->results_status() eq "failed" ) {
		my $msg = "Error connecting to $filer - invalid authentication or
filername?";
		$msg   .= "\nAPI result: " . $results->results_reason();
		logmsg($msg,1);
		exit(1);
	} else {
		my $version = $results->child_get_string("version");
		$version =~ s/NetApp\s+Release\s+(\S+):.*/$1/;
		logmsg("Connectivity OK - ONTAP version $version.",0);
	}

	return 1;
}

# Subroutine to start the consistency group snapshot.
sub cgstart {
	my ($filer) = (@_);
	my ($volume,$naElem,$volElem,$results);

	logmsg("Running cg-start API for filer $filer.",0);
	$naElem = NaElement->new("cg-start");
	$naElem->child_add_string("snapshot",$snapshot);
	$volElem = NaElement->new("volumes");
	for $volume (keys %{ $volumes{$filer} }) {
		logmsg("    -> Including volume $volume.",0);
		$volElem->child_add_string("volume-name",$volume);
	}
	$naElem->child_add($volElem);
	$results = $zapiServer{$filer}->invoke_elem($naElem);
	if ($results->results_status() eq "failed") {
		my $msg = "Error starting CG snapshot for $filer: " . $results-
>results_reason();
		logmsg($msg,1);
		exit(1);
	} else {
		$cgid{$filer} = $results->child_get_string("cg-id");
	}

	return 1;
}

# Subroutine to start the consistency group snapshot.
sub cgcommit {
	my ($filer) = (@_);
	my ($results);

	logmsg("Running cg-commit API for filer $filer with cg-id $cgid{$filer}.",0);
```

```
	$results = $zapiServer{$filer}->invoke("cg-commit","cg-id",$cgid{$filer});
	if ($results->results_status() eq "failed") {
		my $msg = "Error committing CG snapshot for $filer:$volume: " . $results-
>results_reason();
		logmsg($msg,1);
		exit(1);
	}

	return 1;
}

# Help subroutine.
sub help {
	my ($error) = (@_);
	print <<END;

ERROR: $error

Version: $version
Usage: cgsnap snapshot_name filer|vfiler:volume [filer|vfiler:volume ...]

END
	exit 1;
}
```

CRSNAP.SH

This is a shell script that calls compiled cgsnap.pl and uses a number stored in the file ".cursnap" to generate a unique snapshot name every time it is called.

```
#!/bin/bash

[ -f .cursnap ] || echo "0" > .cursnap

CUR_SNAP_NBR=$(( $(cat .cursnap) + 1 ))
echo $CUR_SNAP_NBR > .cursnap
SNAPNAME=cgsnap_PADB_$CUR_SNAP_NBR
echo -e "Current backup number is $CUR_SNAP_NBR (stored in .cursnap).\nCurrent snap
name is $SNAPNAME\n"

export CGSNAP_VERBOSE=1
./cgsnap $SNAPNAME stlfas3270-9:vol1 stlfas3270-9:vol2 stlfas3270-10:vol1
```

RESSNAP.SH

This is a shell script to perform the steps necessary to restore PADB from the latest snapshot based on the number in .cursnap.

```
#!/bin/bash
#
# this script restores PADB from the latest snap netapp snap by;
# 1. Stopping PADB and the underlying services / directories
# 2. Restore the latest snap
# 3. Remount directories
# 4. Restart PADB flagged to restore local disk from SAN
#
# The paraccel SAN mount points on the compute nodes and the
# paraccel install directory on the leader node must be part
# of the snap.  cqi is a paraccel command to run certain database
# tasks, and also to run programs / scripts across the cluster.
#
# This script is not a ParAccel-supported product, and is provided as-is, without
# warranty of any kind.
```

```
function printTimeFromSeconds()
{
  printf "%02d:%02d:%02d" $(($1/3600)) $(($1/60%60)) $(($1%60))
}

if [ ! -f .cursnap ]; then
  echo "Error.  No .cursnap file"
  exit -1
fi

startTime=`date +%s`
echo -e "Starting Snap Restore\t$(date +"%m/%d/%Y %H:%M:%S")"
CUR_SNAP_NBR=$(cat .cursnap)

SNAPNAME=cgsnap_PADB_$CUR_SNAP_NBR
echo -e "Restoring snap $SNAPNAME\t$(date +"%m/%d/%Y %H:%M:%S")"

echo -e "Stopping PADB and unmounting directories\t$(date +"%m/%d/%Y %H:%M:%S")"
cqi xstop
cqi allx 'sudo service iscsi stop' 2>&1 > /dev/null
cqi stop sysmgr
sudo umount /mnt/paraccel

# do snap restore
echo -e "Restoring From Snapshot \t$(date +"%m/%d/%Y %H:%M:%S")"
ssh root@stlfas3270-9 snap restore -f -t vol -s $SNAPNAME vol1
ssh root@stlfas3270-9 snap restore -f -t vol -s $SNAPNAME vol2
ssh root@stlfas3270-10 snap restore -f -t vol -s $SNAPNAME vol1
echo -e "Restoring From Snapshot \t$(date +"%m/%d/%Y %H:%M:%S")"

echo -e "Mounting directories and starting PADB\t$(date +"%m/%d/%Y %H:%M:%S")"
sudo mount /mnt/paraccel         # remount the paraccel directory
rm ~/padb/data/pg/postmaster.pid        # clean out old postmaster.pid
cqi distbin                      # distribute binaries to computes
cqi start sysmgr 2>&1 > /dev/null# start the padb system manager
cqi allx -qq 'sudo service iscsi start' 2>&1 > /dev/null  # start iscsi (ignore old
pid errors)
cqi restore from san                    # set padb to re-rep from SAN to local disk
cqi xstart                              # start database, begin re-rep process
endTime=`date +%s`
echo -e "Finished Snap Restore\t$(date +"%m/%d/%Y %H:%M:%S")\t$(printTimeFromSeconds
$(($endTime - $startTime)))"
```

WATCHSANRESTORE.SH

Simple shell script to show the number of blocks yet to be replicated. Process is done when count reaches zero.

```
#!/bin/bash
# This script is not a ParAccel-supported product, and is provided as-is, without
# warranty of any kind.
TERM=xterm

# 0: Black, 1: Blue, 2: Green, 3: Cyan, 4: Red, 5: Magenta,6: Yellow, 7: White
# set output color to screen, use white as default
function tfcolor()
{
  c=$1
  if [[ $# -ne 1 || $1 -lt 0 || $1 -gt 7 ]]; then
    c=7
  fi
  tput setf $c
```

```
}

function printTimeFromSeconds()
{
  printf "%02d:%02d:%02d" $(($1/3600)) $(($1/60%60)) $(($1%60))
}
clear

[[ $1 =~ "^[0-9]+$" ]] && sleepSec=$1 || sleepSec=10  # if 1st param is a number, then
use, else 5 sec sleep

lun_io=-1
lun_io=$(psql dev -At -c "select count(*) from stv_underrepped_blocks")

if [ ${lun_io} -le 0 ]; then
  echo -e "No SAN Restore Pending - $(date)\nBye."
  exit 1
fi
start_time=$(date +%s)

tput bold
echo -e "Beginning Restore From SAN: $(tfcolor 3) $(date)$(tfcolor 6)"
tput sc

until [ ${lun_io} -eq 0 ]; do
  lun_io=$(psql dev -At -c "select count(*) from stv_underrepped_blocks")
  echo -en "\t$(date) Number of unreplicated blocks ( $lun_io )"
  sleep ${sleepSec}
  tput el1
  tput rc
done
end_time=$(date +%s)

echo -e "$(tfcolor 7) Finished Restore From SAN: $(tput bold)$(tfcolor 3) $(date)
$(tfcolor 7) ( $(printTimeFromSeconds $((${end_time} - ${start_time})))  )\n"
```

7 REFERENCES

ParAccel Documentation

- The ParAccel Quick Start Installation Guide gives an overview of how to perform the initial installation and setup.
- The ParAccel Administrator's Guide and SQL Reference provides documentation for managing the PADB and as a PADB SQL reference.
- The Release Notes v2.0 provides the release notes for v2.0.x of the PADB software and upgrade instructions.
- The PADB Technical Overview white paper provides an overview of the PADB database and the appliance architecture.
- The Scalable Analytic Appliance: A Technical Overview provides an overview of ParAccel's current data warehousing appliance, which utilizes EMC as the SAN storage.

NetApp Documentation and Resources

- TR-3347: A Thorough Introduction to FlexClone Volumes: http://media.netapp.com/documents/tr-3347.pdf
- TR-3858: Using Crash-Consistent Snapshot Copies as Valid Oracle Backups: http://media.netapp.com/documents/tr-3858.pdf

- TR-3791: Cross-Platform Database Migration Using Oracle Transportable Tablespaces and NetApp FlexClone
  http://media.netapp.com/documents/tr-3791.pdf
- ParAccel Analytic Database for NetApp Data Warehouse Solutions:
- http://media.netapp.com/documents/ds-3091.pdf
- NetApp iSCSI Host Utilities for Linux:
- https://now.netapp.com/NOW/cgi-bin/software/?product=iSCSI+Host+Utilities&platform=Linux

Other

- ESG Lab Validation Report: ParAccel PADB and NetApp SAN Optimized Solution:
  http://www.enterprisestrategygroup.com/media/wordpress/2011/05/ESG-Lab-Validation-Report-NetApp-ParAccel-May-11.pdf

Go further, faster®

www.netapp.com