Technical Report

NetApp EF-Series and Couchbase

Ryan Leonard, NetApp
October 2015 | TR-4462

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

© 2015 NetApp, Inc. All Rights Reserved.

1 INTRODUCTION

The NetApp® EF-Series enables Couchbase environments to maintain the highest levels of performance and uptime for NoSQL workloads even through storage infrastructure fault conditions and ever-changing business requirements. The EF-Series is designed to handle the most extreme application workloads with very low latency. Typical use cases include application acceleration; improving the response time of latency-sensitive applications; and improving the power, environmental, and capacity efficiency of overprovisioned environments. The EF-Series leverages the latest in solid-state-disk (SSD) technologies. It is built on a long heritage of servicing diverse workloads to provide superior business value through acceleration in latency-sensitive and high-I/O environments and providing enterprise-class reliability.

Couchbase is one of the most widely deployed NoSQL instances in the world. With the explosion of Internet use, particularly on mobile devices, with social media uploading, general interaction, and sensor- and machine-generated data, a new paradigm was required. This paradigm needed to move from handling traditional structured databases to handling the new dataset of richer content: variable-length text from tweets and blogs and log data from web servers, images, audio, and video.

Traditional relational databases struggle to deal with this type of largely unstructured content and are generally not malleable enough to react to the degree of diversity required. NoSQL databases, including Couchbase, were designed to be distributed and scale out using a cluster of servers to support the ingest and searchability of this semistructured and structured content.

The NetApp EF-Series provides the performance, capacity, and bullet-proof reliability needed to meet the most demanding Couchbase implementations. This technical report provides an overview of Couchbase, the EF-Series, and the combined architecture, along with the results of workload characterizatons performed on a simulated EF-Series and Couchbase environment. Additional information is provided on the EF-Series configuration and a few best practices along with observations based on this initial test data.

2 Couchbase and Use Cases

Couchbase is a high-performance, distributed-deployment NoSQL server. Generically, Couchbase is memory-centric to provide low latency and high scalability; however, disk requests need to be serviced as quickly as possible. Both Java-Script Object Notation (JSON) documents and binary data can be stored in Couchbase, enabling a wide variety of potential use cases. Several of these cases are discussed in the following section.

2.1 Use Cases

Like other NoSQL alternatives, Couchbase is often deployed for a wide variety of use cases and application types.

- **Product promotion**

  Websites, particularly those for retail consumers, now host a large amount of content for the products on offer, including photos, three-dimensional object models, videos, user reviews, and other rich content. Storing, sorting, and displaying this rich content dynamically with a traditional SQL-oriented database can be a challenge. That is because of the diversity of the content and the need to scale access in a nonlinear manner because of the nature of consumer shopping habits.

- **Real-time analytics**

  Couchbase can be combined with Storm and Hadoop to support real-time analytics. Typically Hadoop Map Reduce jobs are bundled, which doesn't enable real-time use but provides an excellent platform for deeper analysis. For real-time analytics, Storm, combined with a NoSQL instance such as

© 2015 NetApp, Inc. All Rights Reserved.

Couchbase, with its streaming integration, enables the searchability needed in real time to perform actions such as determining trending words and/or phrases or current best sellers.

- **Fraud detection and analysis**

  Fraud detection and analysis is an important subject for any corporation or entity involved in the transfer of money from one location to another for goods and services. Detecting fraud requires a complex set of rules, guidelines, and insight into behaviors, such as where they happen or the time of day they take place, to increase accuracy. This ever-shifting collection of data types can be difficult for a traditional SQL-based database with a fixed schema to accommodate, but a NoSQL application such as Couchbase can do so.

- **Targeted interactions**

  Targeted interactions enable an entity to provide the right data to the right user at the right time. This could be a targeted advertisement based on past browsing behavior; a coupon for a projected future good, need, or service; or suggestions around other content the user might also be interested in. Couchbase can assist with this effort because of its flexible data model that might evolve over time and its capability to process large amounts of data in near real time at high throughput rates.

- **User profiles**

  Similar to targeted interactions, user profiles are associated with engaging with the user. However, in this context these profiles relate more to supporting the user and allowing authentication across diverse platforms. The profiles also relate to supporting items such as rich account profiles that might contain images, videos, or other rich content to improve user engagement.

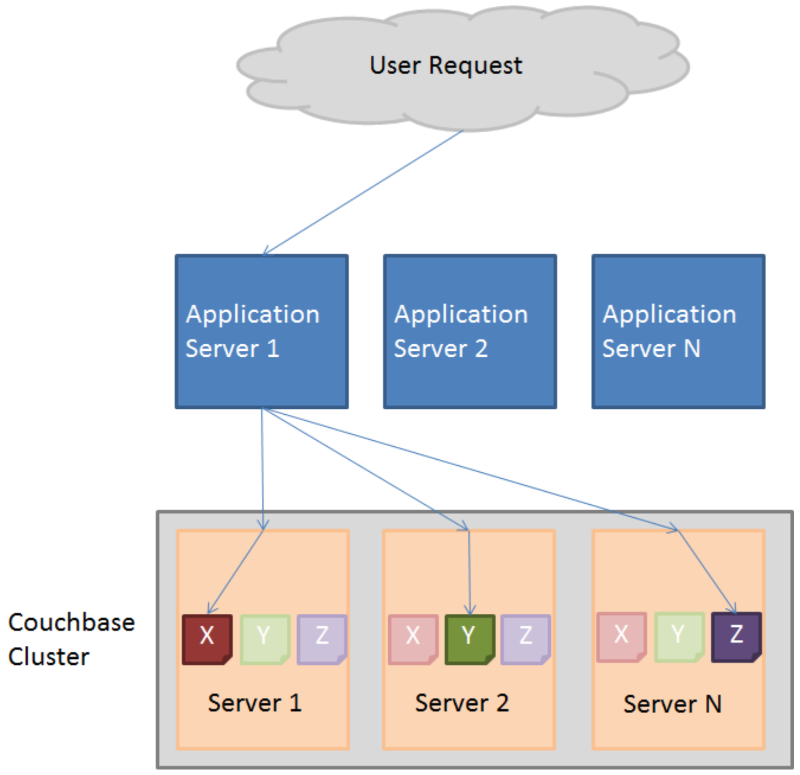
2.2 Architecture

Couchbase is a NoSQL document–oriented database designed to provide fast access to data that requires a flexible data model with the ability to scale. Couchbase is often deployed in a "shared-nothing" architecture among a cluster of servers but it can also be configured to leverage the improved reliability and ease of management provided by external storage.

Within Couchbase, all data objects are stored as either JSON documents or binary data. This data is generally replicated multiple times throughout several nodes in the cluster for improved reliability and access to data if a server fails in the cluster.

Figure 1, shows an example of a user request for documents X, Y, and Z to be processed by Application Server 1. Application Server 1 in turn passes the request to the Couchbase cluster. The server then determines that the primary copy of the desired data for X is on Couchbase Cluster Server 1, Y is on Couchbase Cluster Server 2, and Z is on Couchbase Cluster Server N. Two other copies for each document also reside in the cluster and could be promoted to the primary if any of the cluster nodes fail. In this way, Couchbase provides additional tolerance in a cluster server failure.

Figure 1) Couchbase cluster architecture.



Within Couchbase, documents are placed evenly across all servers in the cluster. Containers called "buckets" are used for data storage, and different buckets might have different replica and caching policies. Buckets can be replicated up to three times across the Couchbase cluster. Each bucket in the Couchbase cluster is split into 1,024 partitions known as "vBuckets" that are mapped across the cluster. In this way, each cluster server manages only a portion of the total vBuckets across the cluster. Some of the vBuckets are active and the others are replicas, if configured, in the event of a cluster server failure. If a failure occurs, one of the replica vBuckets is promoted to active status to handle inbound I/O to the contents of that vBucket.

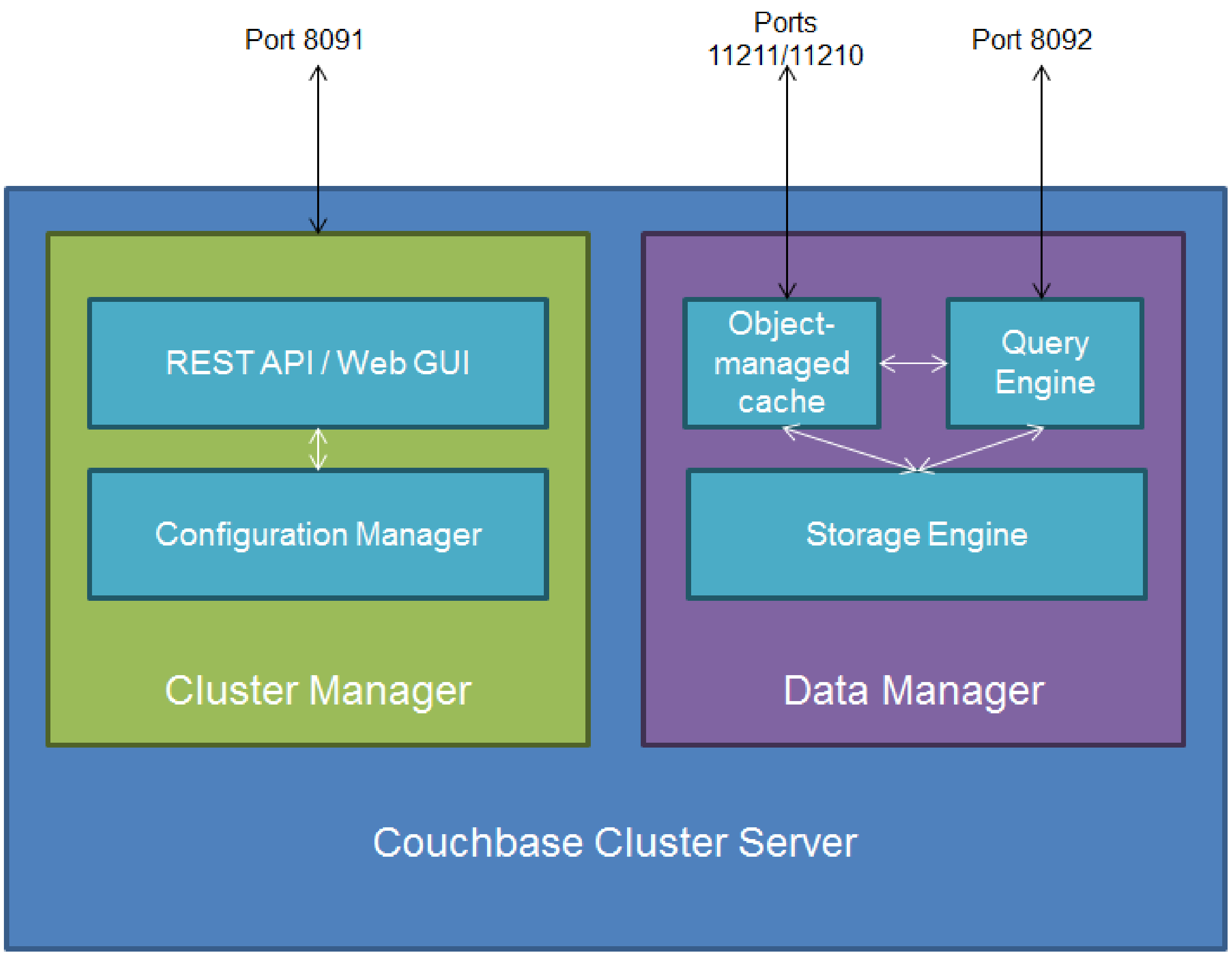
When a document is placed into the cluster, the Couchbase smart client residing on the client server hashes it using CRC32, a polynomial division-based 32-bit cyclic redundancy check of the document's binary contents. The result is used as a document identifier. This identifier is used with the Couchbase "cluster map" to track the location of all vBuckets and their contents so that subsequent lookups can more readily find the correct document when requested.

In order for a Couchbase client application server to interact with the Couchbase cluster, it must use a memcached compatible smart client solution developers kit (SDK). Such SDKs are available in a number of different programming languages, including C, C++, Java, Ruby, and several others. A memcached system is a distributed memory object caching system. When one is employed with Couchbase, the clients become cluster aware and receive updates on the cluster map. This action enables each client to communicate directly with the appropriate cluster node containing the document being requested. After a Couchbase client accesses the Couchbase cluster, the cluster map is requested and transferred, and the client maintains a streaming connection to the cluster to receive updates.

Within Couchbase, all updates are done at the document level. In Couchbase, by default, all client write requests are asynchronous in nature and are considered complete once they are added to the cache of the destination Couchbase cluster server. At a later point in time, these new or updated documents will be replicated, if desired, to other servers in the cluster. Following that point, they will be persisted to the underlying disk media. In this way, Couchbase is a server memory–centric architecture through which most requests will or might be serviced from memory. If a document is not held in memory, fast access to the underlying disk media is imperative to provide high-level quality of service to the requesting client.

Couchbase servers have two major components: the Data Manager and the Cluster Manager. The Data Manager is responsible for underlying management of the documents stored within the database and the Cluster Manager is responsible for managing the cluster. Within the Data Manager are three major subcomponents: the object-managed cache, the storage engine, and the query engine. The relationship beween these components is shown in Figure 2.

Figure 2) Couchbase server major components.



2.2.1 Object-Managed Cache

The object-managed cache of a Couchbase cluster server contains the document identifier, metadata related to the document, and, in some cases, the document itself. As mentioned previously, memcached compatible APIs are required for the client(s) to access the object-managed cache. If a client is not able to use memcached compatible APIs such as GET, SET, and DELETE, then it is still possible to interact with a Couchbase cluster using a memcached proxy such as Moxi. For smart clients using the API natively, communication would be with port 11210 to the object-managed cache; with proxy clients, communication would be with port 11211.

The object-managed cache within the cluster server has a number of defined vBuckets, each with its own associated hashtable. Couchbase implements a locking mechanism such that the hashtable for each is subdivided, enabling multiple workers to access specific sections of the hashtable. By default, three worker threads are created per vBucket: two for reads and one for writes, although this limit can be increased to a total of eight.

To track changes to documents in the cluster, a linked list data structure called a “checkpoint” is created. Documents that have not been persisted to physical disk or replicated to another replica vBucket (if configured) are tracked in the checkpoint. If a mutation of a stored document exists within Couchbase, then only the latest changes are asynchronously persisted to physical disk, providing native deduplication.

2.2.2 Storage Engine

Couchbase uses an append-only design, meaning that as documents are modified, the changes are appended to the end of the existing file and there are no in-place updates. This approach enables write performance improvements when documents are persisted to physical media because all data is written sequentially. A process called compaction is periodically used to copy the nonstale portions of a data file to a new data file to reclaim storage capacity. This compaction cycle is initiated when the exisitng file size reaches a point that exceeds the actual file size by a ratio configured through the Couchbase administrative interface.

2.2.3 Query Engine

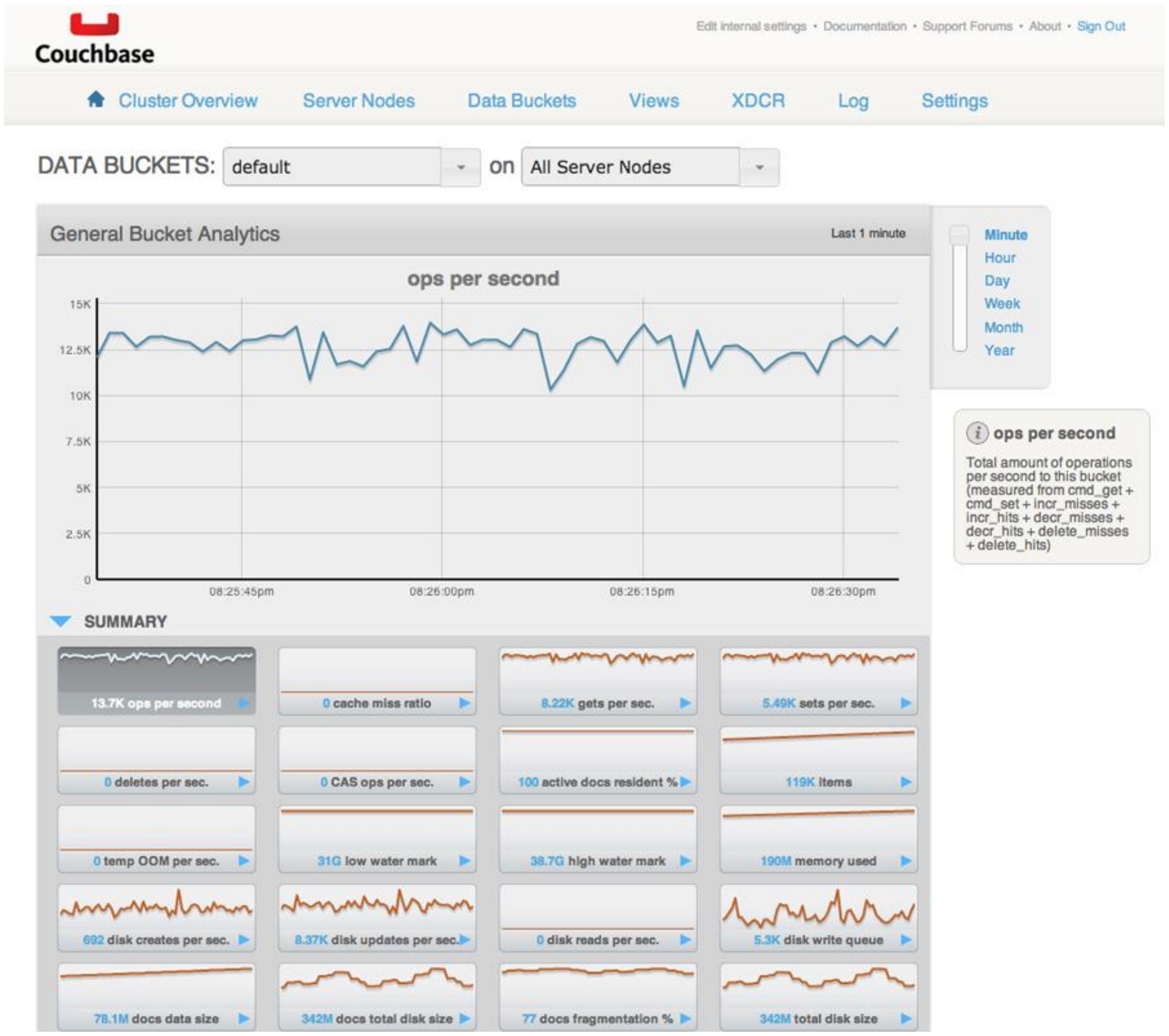
The JSON documents stored with a Couchbase cluster can be queried and are indexed, with secondary indexing created using items known as “design documents” and “views.” Each defined design document can have several views that are processed sequentially and effectively can perform a map with an optional reduce function against the incoming query. The index in Couchbase is built using the attributes as specified in the map function. The index is created by each cluster server for the active vBucket(s) as well as potentally any replica vBuckets on the server. Views in Couchbase occur using the concept of eventual indexing, meaning that indexes are not necessarily updated when documents are modified but instead are update at query time. Similar to the way documents are handled in Couchbase, the views are created in an append-only fashion and have their own compaction procedures as well.

Application servers can run queries against the Couchbase database using port 8092, as shown in Figure 2.

2.2.4 Cluster Manager

The Cluster Manager in Couchbase manages all cluster operations, such as rebalancing because of removal or failure of a cluster failure and replication of vBuckets between Couchbase cluster servers. The Cluster Manager also provides administrative control and management through REST APIs and/or a web-based GUI (shown in Figure 3).

Figure 3) Couchbase web-based GUI.



ops per second

Total amount of operations per second to this bucket (measured from cmd_get + cmd_set + incr_misses + incr_hits + decr_misses + decr_hits + delete_misses + delete_hits)

13.7K ops per second
0 cache miss ratio
8.22K gets per sec.
5.49K sets per sec.
0 deletes per sec.
0 CAS ops per sec.
100 active docs resident %
119K items
0 temp OOM per sec.
31G low water mark
38.7G high water mark
190M memory used
692 disk creates per sec.
8.37K disk updates per sec.
0 disk reads per sec.
5.3K disk write queue
78.1M docs data size
342M docs total disk size
77 docs fragmentation %
342M total disk size

All servers within the cluster run the Cluster Manager. However, within the Couchbase cluster the cluster elects an “orchestrator node” to provide cluster-wide services such as determining when another server in the cluster is unavailable. If the orchestrator node becomes unavailable, the remainder of the cluster elects a new orchestrator node once the heartbeat expires.

3 NetApp EF-Series Overview

The NetApp EF560 all-flash array provides a robust platform for delivering exceptional performance to mission-critical applications such as Couchbase. The EF560 all-flash array leverages the latest in solid-state-disk technologies and a strong heritage of handling diverse workloads to provide superior business value through accelerating latency-sensitive and high-I/O environments.

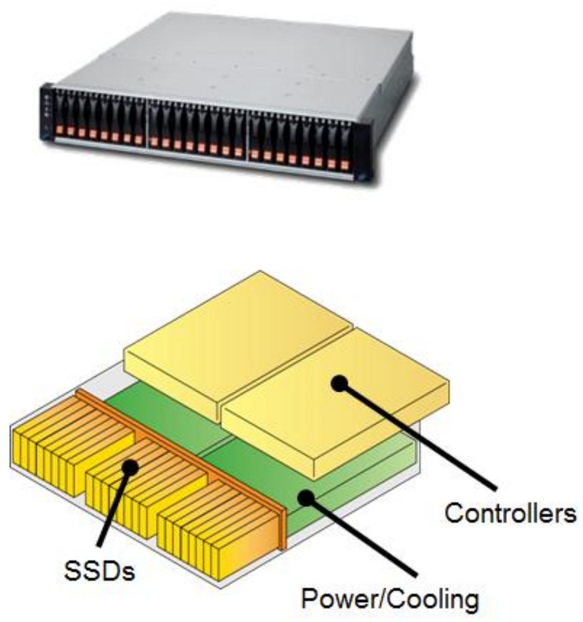
The EF560 is available with up to 192TB of raw SSD-based storage and provides the capacity and bullet-proof reliability needed to meet the requirements of the most demanding organizations. In a recent IDC opinion white paper, the reliability metrics of over 1,000 deployed EF-Series systems were tracked through the NetApp AutoSupport™ (ASUP®) function. For additional information on how the EF-Series provides greater than six-9s availability to the vast majority of the deployed field population, please contact your local NetApp representative for a copy of the IDC document.

3.1 EF560 Hardware Overview

High-performance applications and platforms require reliable, robust, and flexible storage systems to meet demanding requirements. The EF560 all-flash array meets these requirements by supporting:

- Multiple host interface cards for host access, one per controller:
  - Four-port 16Gb FC
  - Two-port 40Gb IB
  - Four-port 10Gb iSCSI
  - Four-port 12Gb SAS
- Up to 120 SSDs per storage system
- Multiple RAID levels (0, 1, 10, 3, 5, and 6) and support for Dynamic Disk Pools (DDP)
- Optional data assurance capabilities (T10PI data integrity checking)
- Media parity check and correction capability
- Extensive event logging
- Recovery Guru on-board system diagnostics and recovery capability
- Hardware redundancy
- 12GB cache memory per controller to maximize read/write performance
- NVSRAM and on-board USB drive to preserve the system configuration during power outages

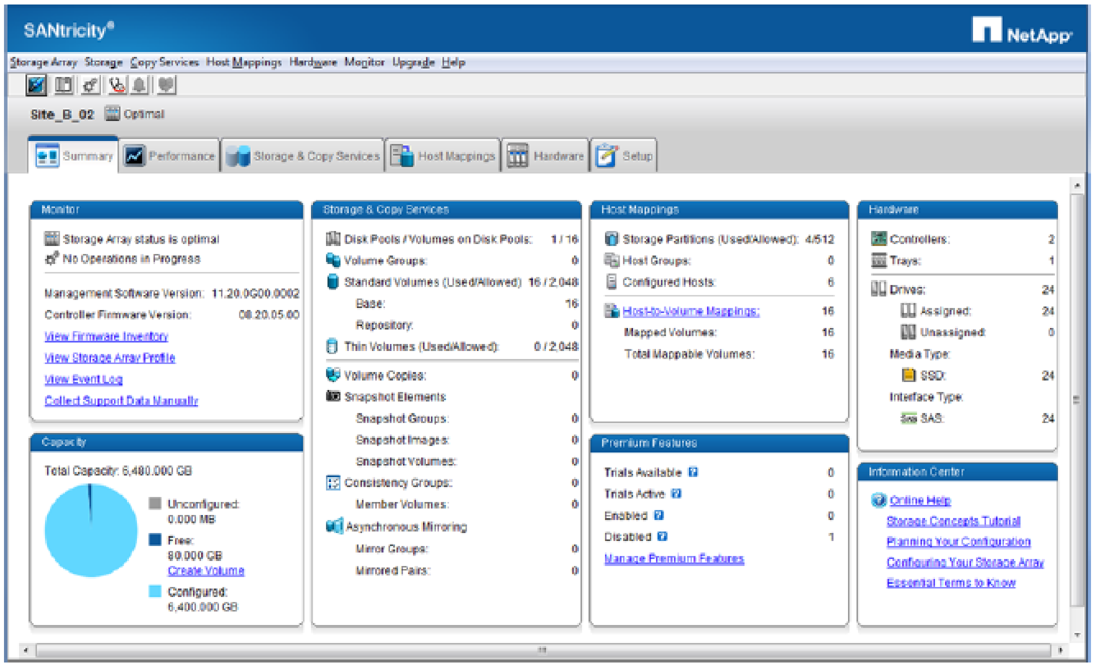
Figure 4) EF560 hardware.



3.2 SANtricity

NetApp SANtricity® software is the GUI management interface for the EF-Series that is based on the Java framework. SANtricity can be installed on Windows or Linux operating systems. The management application must be installed on a management node that does not participate in production data delivery. The software is available in 32-bit and 64-bit versions, and the installation process detects whether installation of the package was performed on the incorrect OS version.

Figure 5) SANtricity Array Management Window Summary tab.



Site_B_02 Optimal

Monitor

Storage Array status is optimal
No Operations in Progress

Management Software Version:	11.20.0G00.0002
Controller Firmware Version:	08.20.05.00

View Firmware Inventory
View Storage Array Profile
View Event Log
Collect Support Data Manually

Capacity

Total Capacity: 6,480.000 GB

Unconfigured: 0.000 MB
Free: 80.000 GB
Create Volume
Configured: 6,400.000 GB

Storage & Copy Services

Disk Pools / Volumes on Disk Pools:	1 / 16
Volume Groups:	0
Standard Volumes (Used/Allowed)	16 / 2,048
Base:	16
Repository:	0
Thin Volumes (Used/Allowed):	0 / 2,048
Volume Copies:	0
Snapshot Elements	
Snapshot Groups:	0
Snapshot Images:	0
Snapshot Volumes:	0
Consistency Groups:	0
Member Volumes:	0
Asynchronous Mirroring	
Mirror Groups:	0
Mirrored Pairs:	0

Host Mappings

Storage Partitions (Used/Allowed):	4/512
Host Groups:	0
Configured Hosts:	6
Host-to-Volume Mappings:	16
Mapped Volumes:	16
Total Mappable Volumes:	16

Premium Features

Trials Available	0
Trials Active	0
Enabled	0
Disabled	1

Manage Premium Features

Hardware

Controllers:	2
Trays:	1
Drives:	24
Assigned:	24
Unassigned:	0
Media Type:	
SSD:	24
Interface Type:	
SAS:	24

Information Center

Online Help
Storage Concepts Tutorial
Planning Your Configuration
Configuring Your Storage Array
Essential Terms to Know

© 2015 NetApp, Inc. All Rights Reserved.

SANtricity client software can be installed on Windows or Linux operating systems for out-of-band management of the storage array. In this configuration, the host agent functionality for in-band management does not function, and the number of client connections is limited to eight. To manage the storage arrays by using in-band connections, the management client must run on a server OS and have Fibre Channel (FC) connectivity to all arrays. The eight-connection limit for out-of-band management client connections does not apply to in-band management.

To create volume groups on flash arrays, the first step when configuring SANtricity is to assign a redundant array of inexpensive disks (RAID level). This assignment is then applied to the disks selected to form the volume group.

The EF560 all-flash arrays support RAID levels 0, 1, 3, 5, 6, and 10 or Dynamic Disk Pools (DDP).

To simplify the storage provisioning, NetApp provides a SANtricity automatic configuration feature. The configuration wizard analyzes the available disk capacity on the array. It then selects disks that maximize array performance and fault tolerance while meeting capacity requirements, hot spares, and any other criteria specified in the wizard.

3.2.1 Dynamic Capabilities

From a management perspective, SANtricity offers a number of capabilities to ease the burden of storage management, including the following:

- New volumes can be created and are immediately available for use by connected servers.
- New RAID sets (volume groups) or Dynamic Disk Pools can be created any time from unused disk devices.
- Volumes, volume groups, and disk pools can all be expanded online as necessary to meet any new requirements for capacity or performance.
- Dynamic RAID migration allows the RAID level of a particular volume group, for example, from RAID 10 to RAID 5, to be modified online if new requirements dictate a change.
- Flexible cache block and segment sizes enable optimized performance tuning based on a particular workload. Both items can also be modified online.
- There is built-in performance monitoring of all major storage components, including controllers, volumes, volume groups, pools, and individual disk drives.
- Automated remote connection to the NetApp AutoSupport function enables “phone home” capabilities and automated parts dispatching in case of component failures.
- Path failover and load-balancing (if applicable) between the host and the redundant storage controllers in the EF560 are provided.
- You can manage and monitor multiple EF-Series and/or E-Series storage systems from the same management interface.

3.2.2 Dynamic Disk Pools

With seven patents pending, the DDP feature dynamically distributes data, spare capacity, and protection information across a pool of disk drives. These pools can range in size from a minimum of 11 drives to as many as all the drives in an EF560 storage system. In addition to creating a single DDP, storage administrators can opt to create traditional volume groups with a single DDP or even multiple DDPs, which offer an unprecedented level of flexibility.

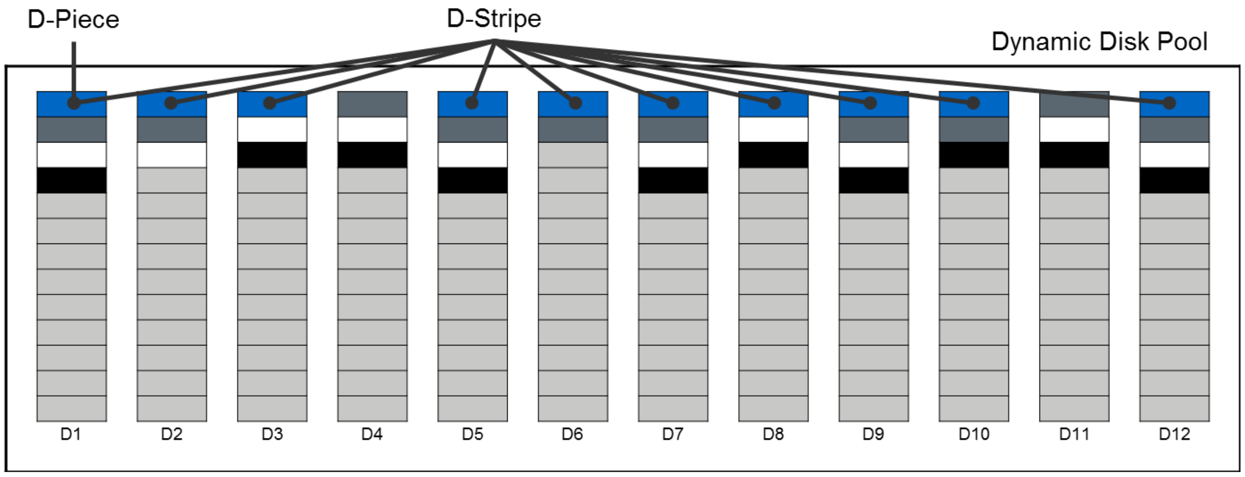
Dynamic Disk Pools are composed of several lower-level elements. The first of these is a D-piece. A D-piece consists of a contiguous 512MB section from a physical disk that contains 4,096 128KB segments. Within a pool, 10 D-pieces are selected using an intelligent optimization algorithm from selected drives

© 2015 NetApp, Inc. All Rights Reserved.

within the pool. Together, the 10 associated D-pieces are considered a D-stripe, which is 4GB of usable capacity in size. Within the D-stripe, the contents are similar to a RAID 6 8+2 scenario. There, 8 of the underlying segments potentially contain user data, 1 segment contains parity (P) information calculated from the user data segments, and 1 segment contains the Q value as defined by RAID 6.

Volumes are then created from an aggregation of multiple 4GB D-stripes as required to satisfy the defined volume size up to the maximum allowable volume size within a DDP. Figure 6 shows the relationship between these data structures.

Figure 6) Dynamic Disk Pool components.

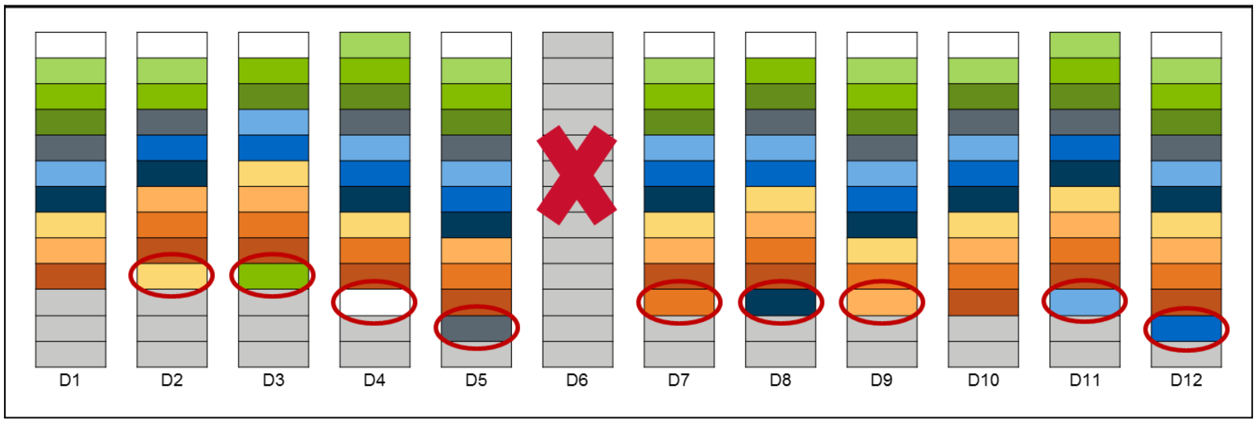


Another major benefit of a DDP is that, rather than using dedicated stranded hot spares, the pool contains integrated preservation capacity to provide rebuild locations for potential drive failures. This approach simplifies management, because individual hot spares no longer need to be planned or managed. The approach also greatly improves the time for rebuilds, if required, and enhances volume performance during a rebuild, as opposed to traditional hot spares.

When a drive in a DDP fails, the D-pieces from the failed drive are reconstructed to potentially all other drives in the pool using the same mechanism normally used by RAID 6. During this process, an algorithm internal to the controller framework verifies that no single drive contains two D-pieces from the same D-stripe. The individual D-pieces are reconstructed at the lowest available LBA range on the selected disk drive.

© 2015 NetApp, Inc. All Rights Reserved.

Figure 7) Dynamic Disk Pool drive failure.



In Figure 7, disk drive 6 (D6) is shown to have failed. Subsequently, the D-pieces that previously resided on that disk are recreated simultaneously across several other drives in the pool. Because there are multiple disks participating in the effort, the overall performance impact of this situation is lessened and the length of time needed to complete the operation is dramatically reduced.

When multiple disk failures occur within a DDP, priority for reconstruction is given to any D-stripes missing two D-pieces to minimize data availability risk. After those critically affected D-stripes are reconstructed, the remainder of the necessary data is reconstructed.

From a controller resource allocation perspective, there are two user-modifiable reconstruction priorities within DDP:

- Degraded reconstruction priority is assigned to instances in which only a single D-piece must be rebuilt for the affected D-stripes; the default for this is high.
- Critical reconstruction priority is assigned to instances in which a D-stripe has two missing D-pieces that need to be rebuilt; the default for this is highest.

For very large disk pools with two simultaneous disk failures, only a relatively small number of D-stripes are likely to encounter the critical situation in which two D-pieces must be reconstructed. As discussed previously, these critical D-pieces are identified and reconstructed initially at the highest priority. Doing so returns the DDP to a degraded state quickly so that further drive failures can be tolerated.

In addition to improving rebuild times and providing superior data protection, DDP can also greatly improve the performance of the base volume when under a failure condition compared to the performance of traditional volume groups.

3.3 Performance

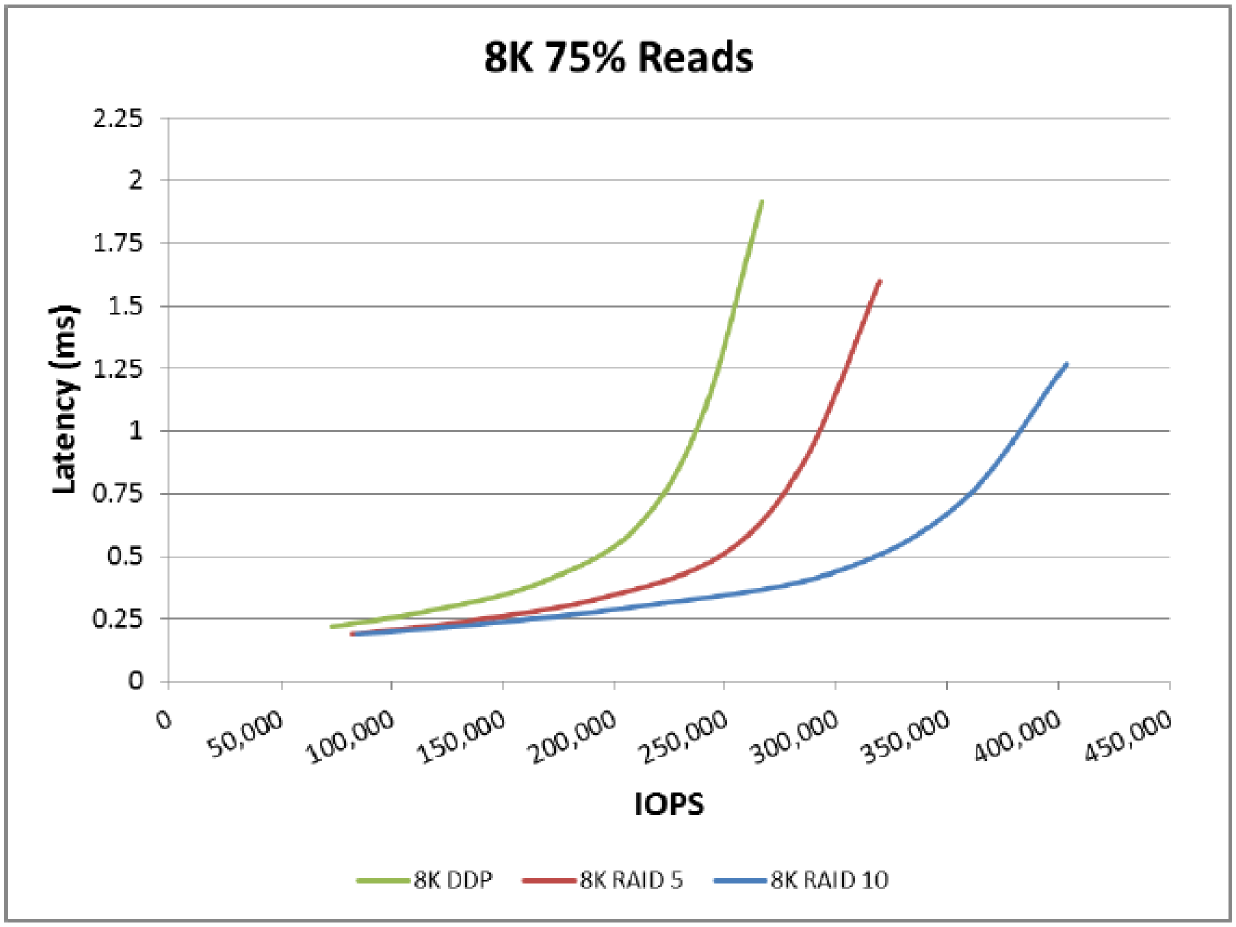
As an all-flash array, the EF560 can perform at very high levels, both in I/Os per second (IOPS) and throughput, while still providing extremely low latency. In many Couchbase deployments, internal solid-state disk drives located in the server are often deployed to provide the level of performance required of an expected high-performance NoSQL cluster. The EF560, through its ease of management, higher degree of reliability, and exceptional performance, can meet the extreme performance requirements expected when a document is not located in the memory of a Couchbase cluster server.

An EF560 is capable of providing up to 650,000 4KB random read IOPS at less than a 1ms average response time. It can provide up to 100,000 4KB random write IOPS at less than a 1ms average response time when configured with RAID 5.

Many factors can affect the performance of the EF560, including different volume group types or the use of DDP, the average I/O size, and the read versus write percentage provided by the attached server(s).

Figure 8 provides performance statistics across various data protection strategies using a larger 8KB I/O size with 75% reads on the system under a generic I/O workload.

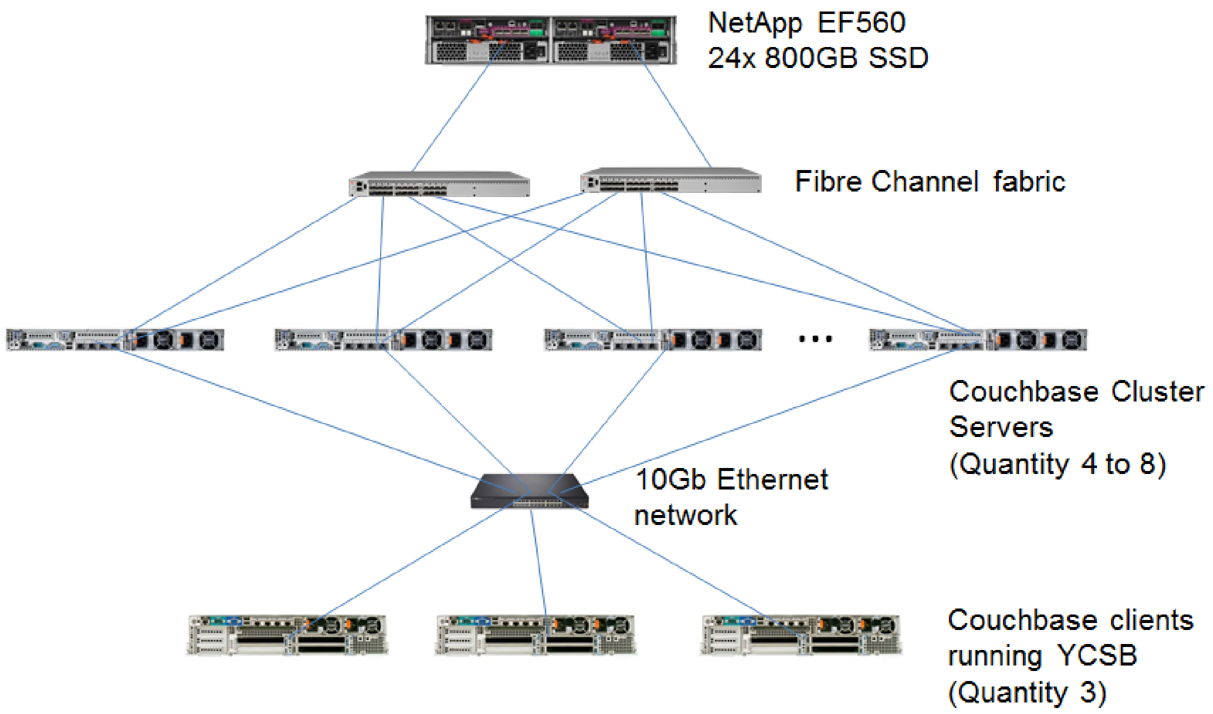
Figure 8) EF560 performance.



4 Couchbase and EF560 Testing

NetApp recently tested a simulated Couchbase environment with the EF560. The environment used a multiserver Couchbase cluster and several client servers using the Yahoo Cloud Serving Benchmark (YCSB) to generate appropriate workloads for the combined architecture. The tested environment is depicted in Figure 9.

Figure 9) Couchbase and EF560 test environment.


NetApp EF560
24x 800GB SSD
Fibre Channel fabric
Couchbase Cluster Servers (Quantity 4 to 8)
10Gb Ethernet network
Couchbase clients running YCSB (Quantity 3)

From a Couchbase perspective, 200 million documents were created per node, resulting in this data distribution per test:

Table 1) Couchbase configuration.

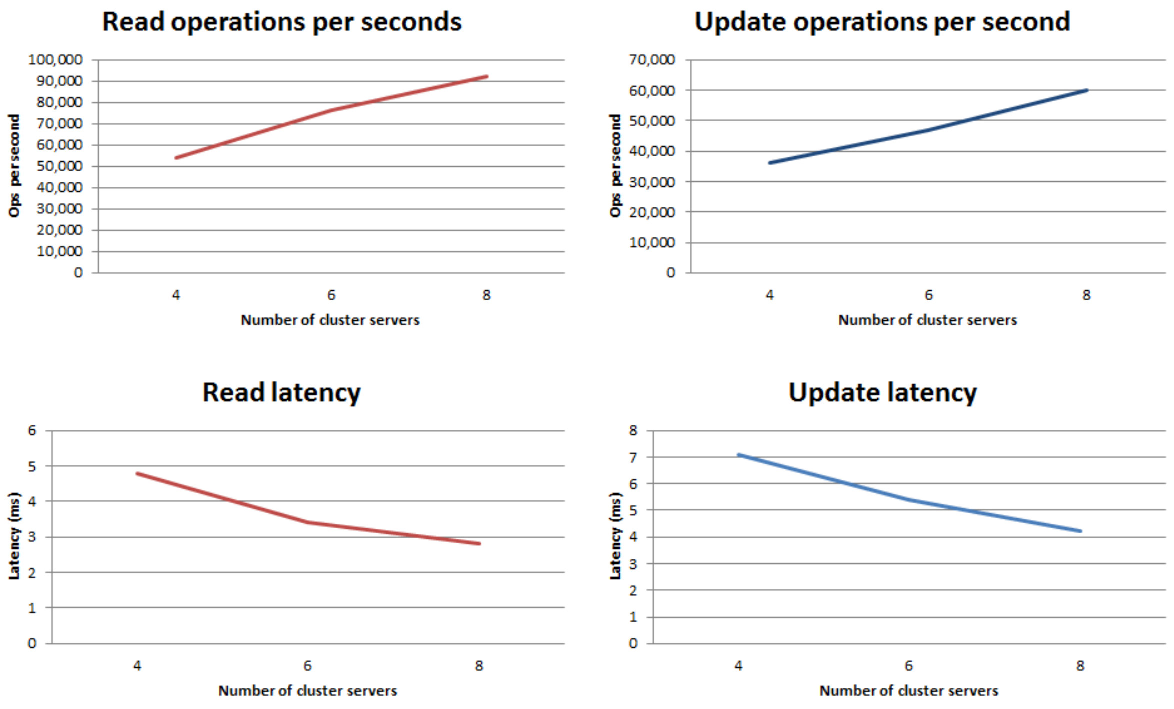
Number of Cluster Servers	Number of Documents	Used Storage	Used RAM	In Memory
4	800M	1.5TB	128GB	16%
6	1.2B	2.2TB	192GB	16%
8	1.6B	3.1TB	256GB	16%

YCSB was configured during all tests as either 100% random reads or 100% random updates to documents within the database using a uniform distribution across the entire dataset. In addition, YCSB target throughput was kept unlimited. These tests were conducted on 4, 6, and 8 cluster servers, respectively. Performance and scalability were tested with a compaction sequence introduced before increasing the number of cluster servers after the update tests were complete.

The EF560 under test was configured with a single DDP of 24 400GB SSDs, with a pool preservation capacity of 2 drives offering ~6TiB of usable capacity. Eight 400GB volumes were created and one volume was presented to each server in the Couchbase cluster.

Figure 10 shows the resulting benchmark results. The operations per second and latency information in these graphs reflects Couchbase collected data as measured by Couchbase at the database level.

Figure 10) Couchbase performance results.

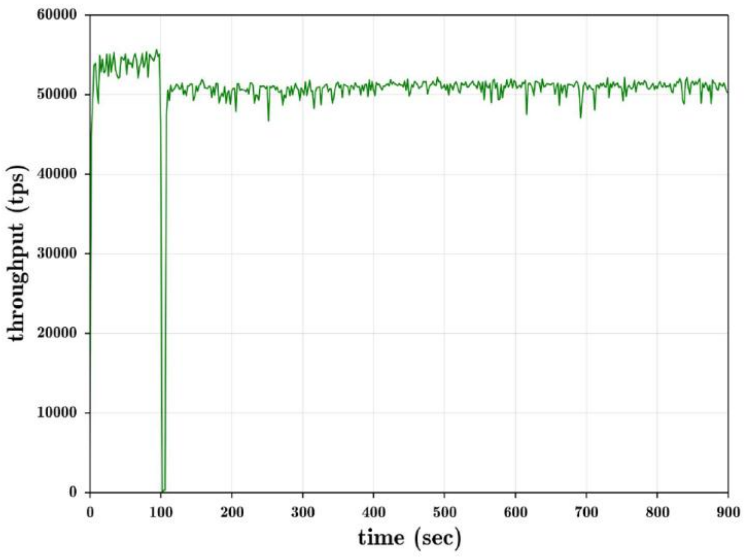


As the graphed data shows, 8 cluster servers produced nearly 100,000 read operations per second using the EF560 and read latency, as measured at the application, of under 5ms. Similarly, for update operations, 8 cluster servers approached 60,000 update operations per second with a maxium update latency of under 8ms. Operations per second for both reads and updates to the Couchbase server documents is near linear, showing that the EF560 can scale with the Couchbase cluster.

In addition to baseline performance testing, additional testing was conducted on the EF560 under failure conditions, including controller failure and drive failure.

In a typical Couchbase deployment using only internal drives within the server, there is no redundancy at the RAID controller level. A failure of the internal controller would be similar to a complete server failure and would require rebalancing of the data across the remaining cluster servers. With the EF560 and the redundancy provided through the dual redundant controller design, no rebalancing is required because all volumes simply transition to the remaining controller. A test was conducted in which a controller within the EF560 was failed under an active workload. Figure 11 shows the result of this test.

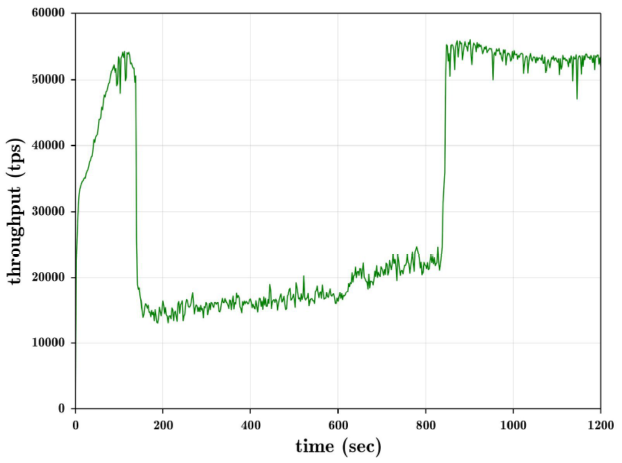
Figure 11) Couchbase performance results, controller failure.



The figure shows a test sequence before, during, and after a controller failure. In this particular test case, the Couchbase cluster server throughput was approximately 55,000 operations per second. A dip can be seen when the surviving controller experiences path failover. Then performance climbs back to approximately 50,000 operations per second because all volumes reside on a single controller rather than being distributed across both controllers in the EF560. The length of time during which the dip occurred was less than 2 seconds.

The final test conducted involved failing a drive within the 24-drive DDP that had been previously created. Figure 12 shows the operations per second that occurred before, during, and after the disk failure.

Figure 12) Couchbase performance results, disk failure.

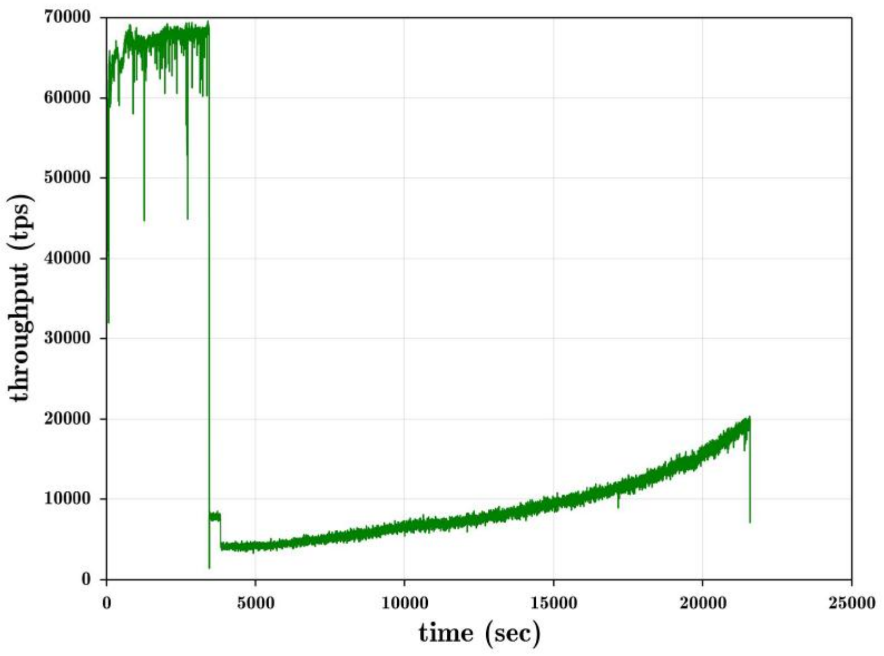


In this case, performance declined from around 55,000 operations per second to 15,000 to 20,000 operations per second. However, this impact was limited in time because Dynamic Disk Pool rebuilds can occur very quickly. Approximately 15 minutes after the initial disk failure, normal service was restored because DDP completed the rebalancing within the surviving members of the disk pool.

© 2015 NetApp, Inc. All Rights Reserved.

A test was also performed on a similar scenario using a server and internal 400GB SSDs configured in a RAID 5, 2+1 configuration. One of the eight cluster servers experienced a simulated drive failure. The results of this fault are shown in Figure 13.

Figure 13) Couchbase performance results, internal disk failure.



In this test, a single drive failure and rebuild process in one of the internal drives in a Couchbase cluster server had a significant impact on the cluster's capability to process requests from clients. The operations-per-second rate dropped by over 90%. In addition to this significant performance impact compared to that of the EF560 using DDP, the length of the impact was drastically extended. In this test case, there were over five hours of collected data and the internal drive rebuild still had not completed.

5 Summary

The NetApp EF560 provides a number of significant advantages over internal drives for Couchbase deployments. These advantages include exceptional storage management capabilities, dramatically improved reliability and high availability, and limited performance degradation because of failure conditions such as disk failures. Another advantage is the excellent performance while handling the most demanding NoSQL workloads with very low latency.

From an administrative standpoint, the EF560 offers simplified storage management with a centralized user interface. This capability enables new volumes, volumes groups, or Dynamic Disk Pools to be created easily and provisioned immediately for use by the Couchbase cluster servers. In addition, existing volumes, volume groups, and Dynamic Disk Pools can be increased in size dynamically to provide additional capacity and/or performance as required for the Couchbase environment.

Copyright Information

Copyright © 1994–2015 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY 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.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark Information

NetApp, the NetApp logo, Go Further, Faster, AltaVault, ASUP, AutoSupport, Campaign Express, Cloud ONTAP, Clustered Data ONTAP, Customer Fitness, Data ONTAP, DataMotion, Fitness, Flash Accel, Flash Cache, Flash Pool, FlashRay, FlexArray, FlexCache, FlexClone, FlexPod, FlexScale, FlexShare, FlexVol, FPolicy, GetSuccessful, LockVault, Manage ONTAP, Mars, MetroCluster, MultiStore, NetApp Insight, OnCommand, ONTAP, ONTAPI, RAID DP, RAID-TEC, SANtricity, SecureShare, Simplicity, Simulate ONTAP, SnapCenter, Snap Creator, SnapCopy, SnapDrive, SnapIntegrator, SnapLock, SnapManager, SnapMirror, SnapMover, SnapProtect, SnapRestore, Snapshot, SnapValidator, SnapVault, StorageGRID, Tech OnTap, Unbound Cloud, WAFL, and other names are trademarks or registered trademarks of NetApp Inc., in the United States and/or other countries. All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such. A current list of NetApp trademarks is available on the web at http://www.netapp.com/us/legal/netapptmlist.aspx. TR-4462-1015

NetApp®
www.netapp.com