



**CASPUR Consorzio Interuniversitario  
per le Applicazioni  
di Supercalcolo  
per Università e Ricerca**

# **Performance evaluation of Linux cluster file system for a large scale e-mail service.**

L. Valcamonici  
([l.valcamonici@caspur.it](mailto:l.valcamonici@caspur.it))

V. Calabritto, M. Filacchioni, G. Foglietta  
December 2004  
Ver. 1.9

## 1. Overview

CASPUR, a computing consortium of Italian Universities, has completed an evaluation of Linux cluster file systems for the deployment of a new email service, Refero. We are primarily concerned with absolute file system performance based on I/O throughput and its ability to serve 800,000 concurrent users. In this application the file system would have to cope with millions of small files (between 0.5KB and 22KB), which is a very challenging and rigorous workload for typical cluster file system architectures.

For this evaluation we chose two performance benchmarks. The first benchmark we used, Postmark 1.5, is an accepted industry standard benchmark which characterizes the I/O performance of workloads similar to email applications. We ran this benchmark as a single process on a single server and as two processes running concurrently on a cluster of two servers. Each test was run with multiple file sizes. The second benchmark we developed internally to simulate operation of IMAP during a “stat” call in the Maildir format. Each benchmark is explained in greater detail in its specific section below.

CASPUR, Consorzio interuniversitario per le Applicazioni di Supercalcolo Per Università e Ricerca, was established in 1992 as an inter-university computing consortium, in Rome, Italy. The Consortium, a non-profit organization, is financed by MIUR (the Ministry for Education, Universities and Research) and by associated Universities. Its purposes are to manage a High Performance Computing (HPC) center, to promote the use of the most advanced information processing systems and to develop research programs aimed at a more effective and innovative usage of information and communication technology.

## 2. Involved hardware

The hardware remained constant for each of the benchmark tests. The specific hardware used is as follows:

### 2 node cluster

- Servers: Dell PE2650, 2x2.8GHz Xeon (533MHz FSB), 2GB RAM, 2x36GB system disks (RAID1), Qlogic 2340 HBA
- Operating System: Linux RHEL 3.0 Update 3 (kernel 2.4.21-15.0.3)

### SAN

- FC switch: Brocade Silk Worm 2400 (8 ports, 1Gb fabric, no zones)
- Storage Array: EMC Clariion CX300, 14x146GB FC spindles (10K RPM) + 1 Hot Spare, single Storage Processor attached to FC switch, single 330GB RAID5 LUN, read+write cache enabled (with mirroring between the two storage processors)

## SAN for IBM TotalStorage SAN File System 2.1 (SANFS)

- FC switch: Qlogic SANBox 5200 (16 ports, 2Gb fabric, no zones)
- 2xIBM DS4500 (FAStT900), 4xEXP100 Storage Expansion Unit, 4x14x250GB SATA spindles (7.2K RPM), parallel use of all four controllers, 4x2TB RAID5 LUNs spanning all spindles, the single 8TB file system use for testing was created by the means of a volume manager collecting all the four LUNs, read+write cache enabled (mirroring disabled between the four controllers).
- 2xIBM dedicated metadata servers.

### 3. Tuning considerations

Before starting the evaluation, we tried various tuning options for all the cluster file system solutions under test. For our particular application (millions of small files), we found the best results (performance and reliability) were achievable with the default software configurations. This is how we ran the benchmarks.

### 4. First test

#### 4.1. Goal

Evaluate the general performance of several Linux cluster file systems for a large scale e-mail service. In this application the file system has to cope with millions of small files (between 0.5KB and 22KB), which is a very challenging and rigorous workload for typical cluster file system architectures.

#### 4.2. The benchmark

We used Postmark 1.5 ([http://www.netapp.com/tech\\_library/3022.html](http://www.netapp.com/tech_library/3022.html)). A description of the benchmark, from the Postmark website, follows:

*PostMark was designed to create a large pool of continually changing files and to measure the transaction rates for a workload approximating a large Internet electronic mail server.*

*PostMark generates an initial pool of random text files ranging in size from a configurable low bound to a configurable high bound. This file pool is of configurable size and can be located on any accessible file system.*

*Once the pool has been created (also producing statistics on continuous small file creation performance), a specified number of transactions occurs. Each transaction consists of a pair of smaller transactions:*

- *Create file or Delete file*
- *Read file or Append file*

*The incidence of each transaction type and its affected files are chosen randomly to minimize the influence of file system caching, file read ahead, and disk level caching and track buffering. This incidence can be tuned by setting either the read or create bias parameters to produce the desired results.*

*When a file is created, a random initial length is selected, and text from a random pool is appended up to the chosen length. File deletion selects a random file from the list of active files and deletes it.*

*When a file is to be read, a randomly selected file is opened, and the entire file is read (using a configured block size) into memory. Either buffered or raw library routines may be used, allowing existing software to be approximated if desired.*

*When all of the transactions have completed, the remaining active files are all deleted (also producing statistics on continuous file deletion).*

Postmark is a single process benchmark, and in this case a single instance of it was started. The chosen workload was 100K files and 100K transactions in a single directory. No “append” on files was allowed (Biases are: read/append=10, create/delete=5). We used Unix buffered file I/O.

Each test was run five times. We discarded the best and the worst results. The data presented herein are the mean values of the three remaining benchmark runs.

Following is a legend of the tests run:

<u>Tps:</u>	Average transaction rate (files/second)
<u>Create/s:</u>	Average file creation rate (files/second)
<u>Create/s with Trans:</u>	Average creation rate (files/second) for files created during sequence of transactions
<u>Read/s:</u>	Average file read rate (files/second)
<u>Delete/s:</u>	Average file deletion rate (files/second)
<u>Delete/s with Trans:</u>	Average deletion rate (files/second) for files deleted during sequence of transactions
<u>Read (KBps):</u>	Over total size of data read, the average input rate (bytes/second)
<u>Write (KBps):</u>	Over total size of data written, the average output rate (bytes/second)

### **4.3. Players**

PSFS: Polyserve Matrix Server 2.6.1

SANFS: IBM TotalStorage San File System 2.1  
 GFS: Red Hat GFS 6.0  
 VXFS: Veritas Cluster File System 4.0  
 Ext3: standard Linux Journal File System (the baseline)

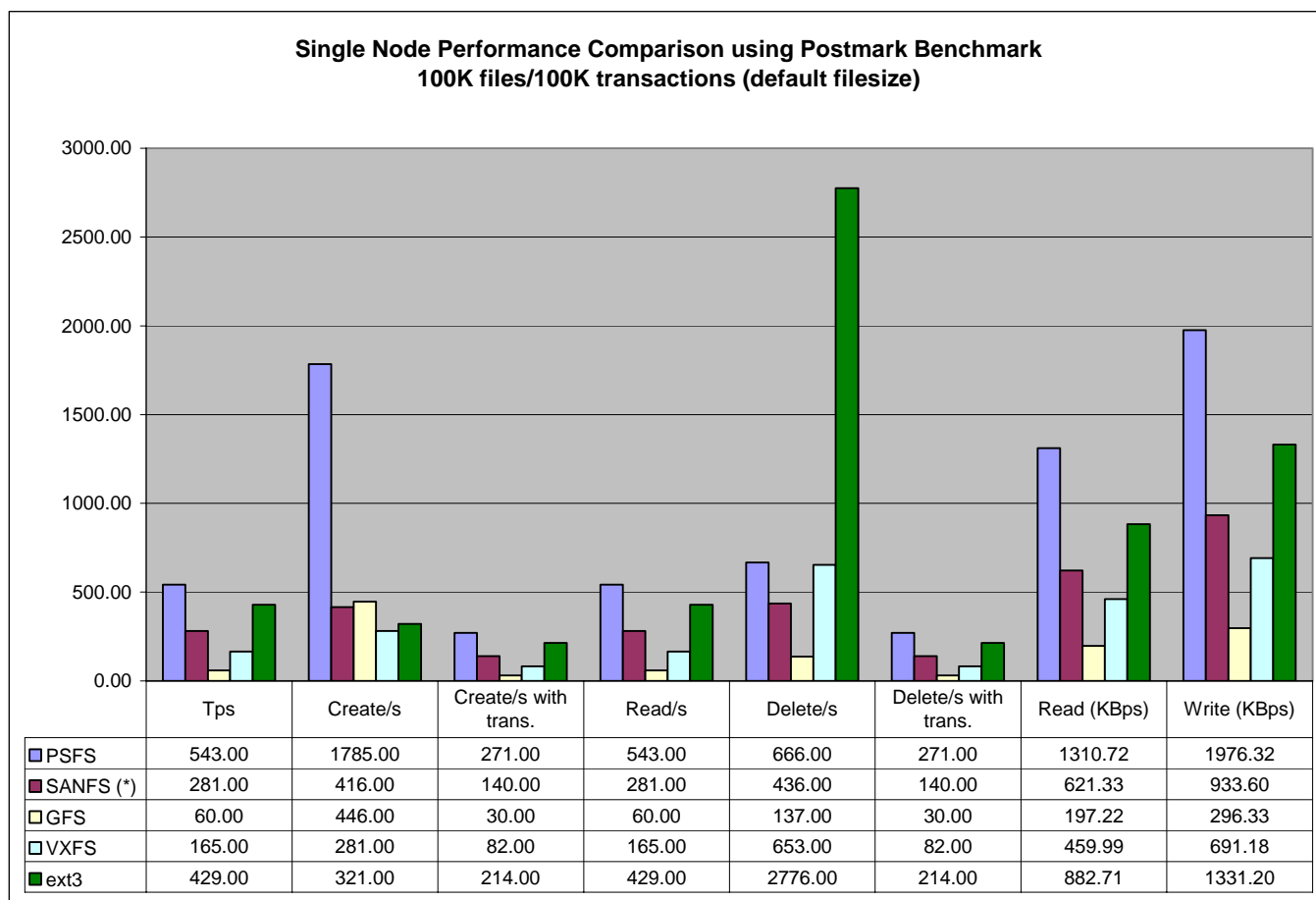
At the beginning ADIC StorNext was also considered, but we decided lately to remove it from the panel because it has a limitation in terms of the maximum number of files the file system can host. The maximum i-node number depends on the file system block size, and with large block size (32KB) we experience a serious decay of overall cluster performance.

#### 4.4. Results

##### First workload:

Default file size (0.5KB-10KB). A single Postmark process ran on a single server, while the other cluster member was idle.

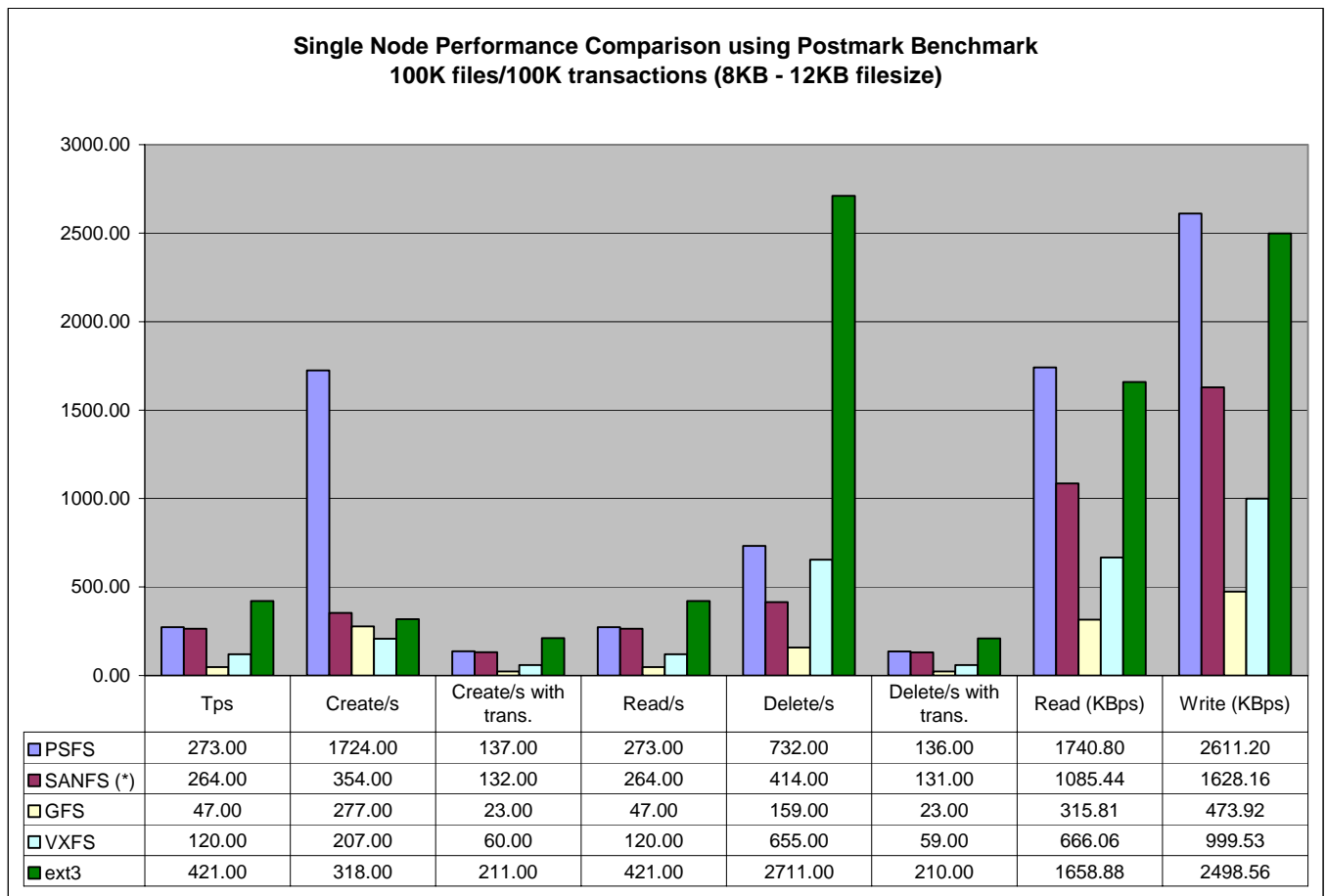
For SANFS, (\*) highlights it used a different SAN configuration.



##### Second workload:

8KB-12KB file size. A single Postmark process ran on a single server, while the other cluster member was idle.

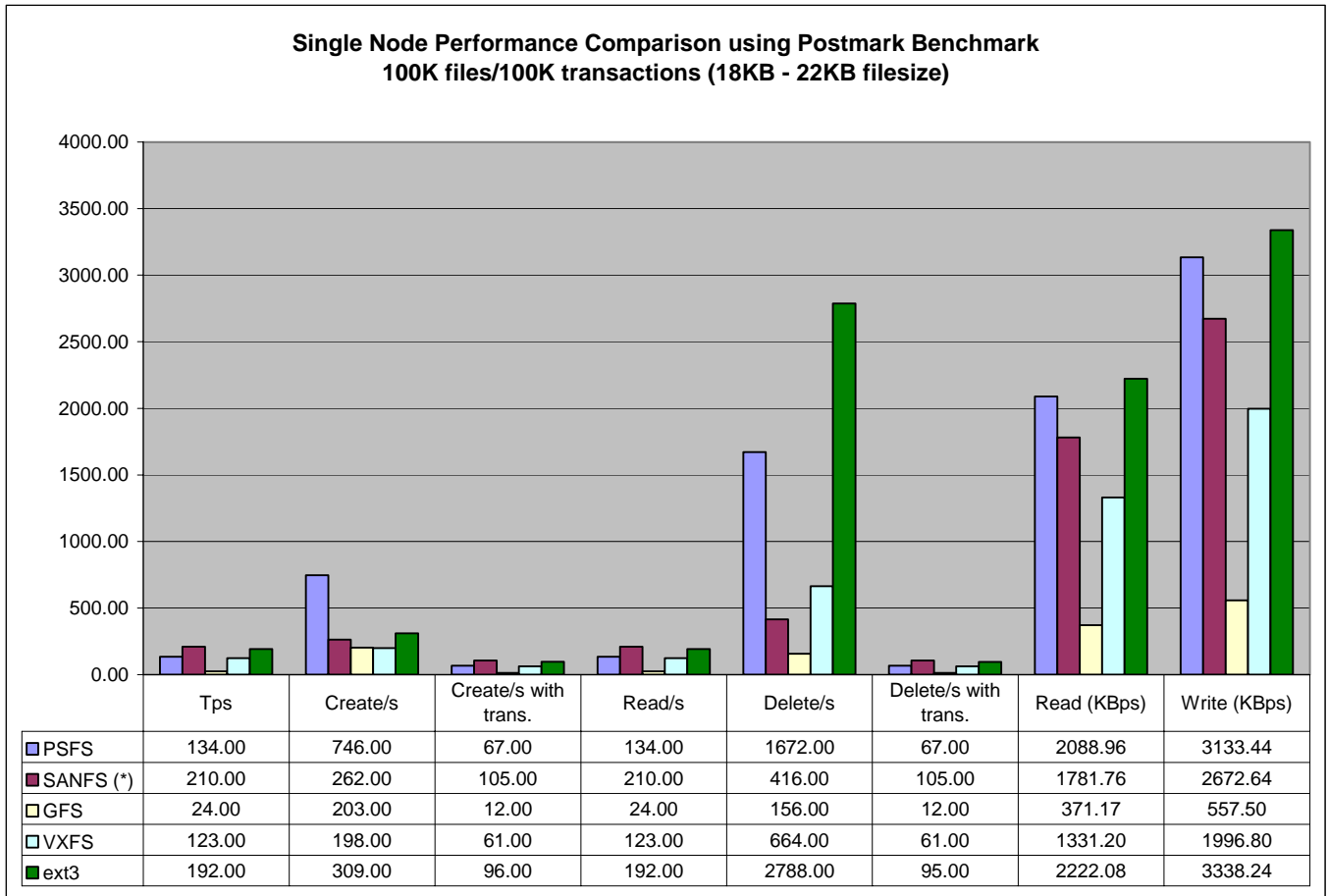
For SANFS, (\*) highlights it was used a different SAN configuration.



Third workload:

18KB-22KB file size. A single Postmark process ran on a single server, while the other cluster member was idle.

For SANFS, (\*) highlights it was used a different SAN configuration.



## **5. Second test**

### **5.1. Goal**

Evaluate the concurrent access performance of the Linux cluster file systems for a large scale e-mail service.

In this application the file system would have to cope with multiple servers having concurrent access to millions of small files (between 0.5KB and 22KB), which is a very challenging and rigorous workload for typical cluster file system architectures.

### **5.2. The benchmark**

We used Postmark 1.5 ([http://www.netapp.com/tech\\_library/3022.html](http://www.netapp.com/tech_library/3022.html)). It is a single process benchmark, and a single instance of it was started concurrently on each of the two cluster members. No “append” on files was allowed (Biases are: read/append=10, create/delete=5). We used Unix buffered file I/O. We chose two different datasets. The first was 100K files and 100K transactions in a single directory for each server. The second carried out half the load: 50K files and 50K transactions in a single directory for each server.

Each test was run five times. We collected the overall results as the sum of the results given by the benchmark running on each server. Then we discarded the best and the worst overall results. The data presented herein are the mean values of the three remaining overall results.

### **5.3. Players**

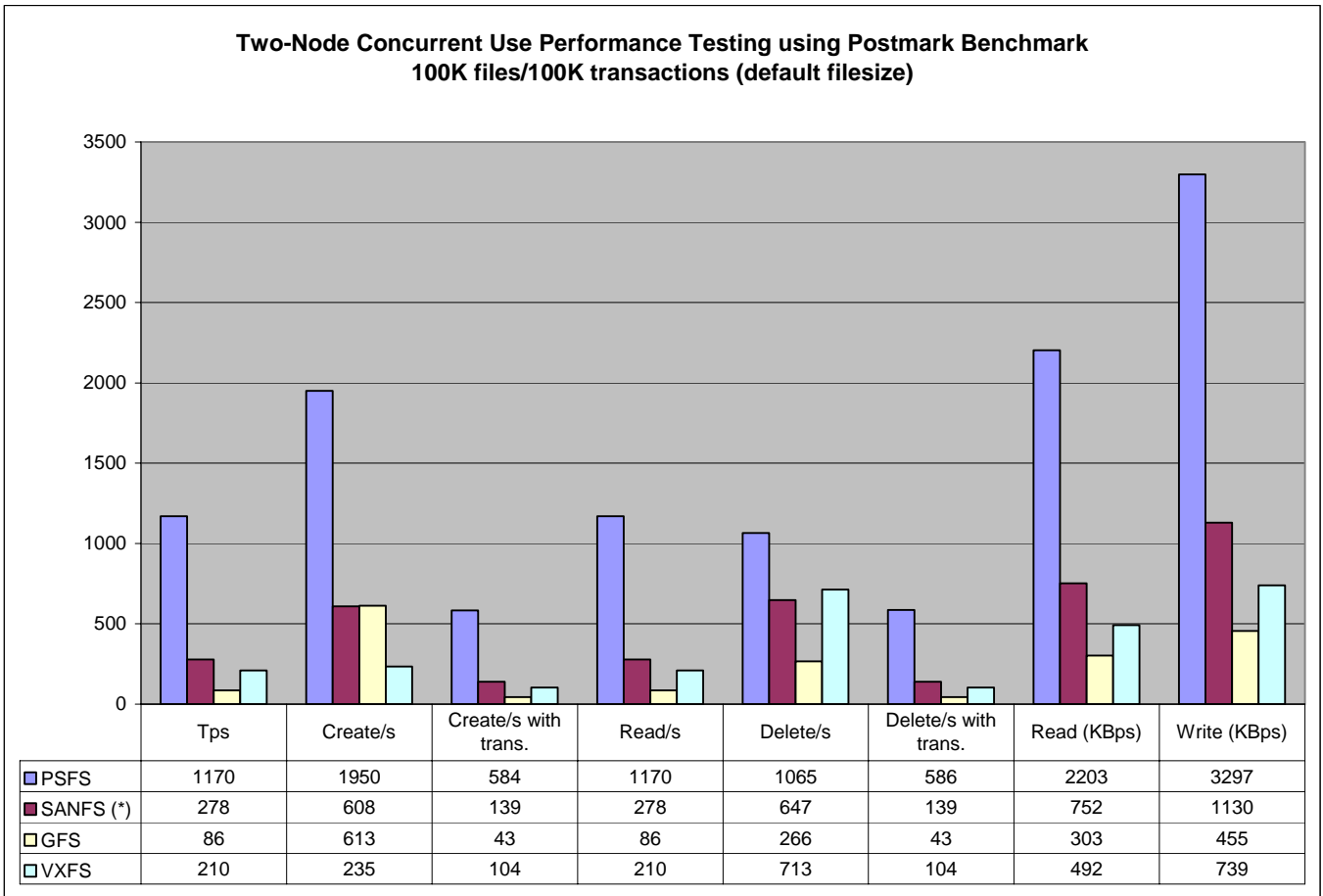
PSFS: Polyserve Matrix Server 2.6.1  
SANFS: IBM TotalStorage San File System 2.1  
GFS: Red Hat GFS 6.0  
VXFS: Veritas Cluster File System 4.0

### 5.4. First dataset

First workload:

A single Postmark process ran on each of the two servers. Each server processed 100K files of default file size (0.5KB-10KB) with 100K transactions.

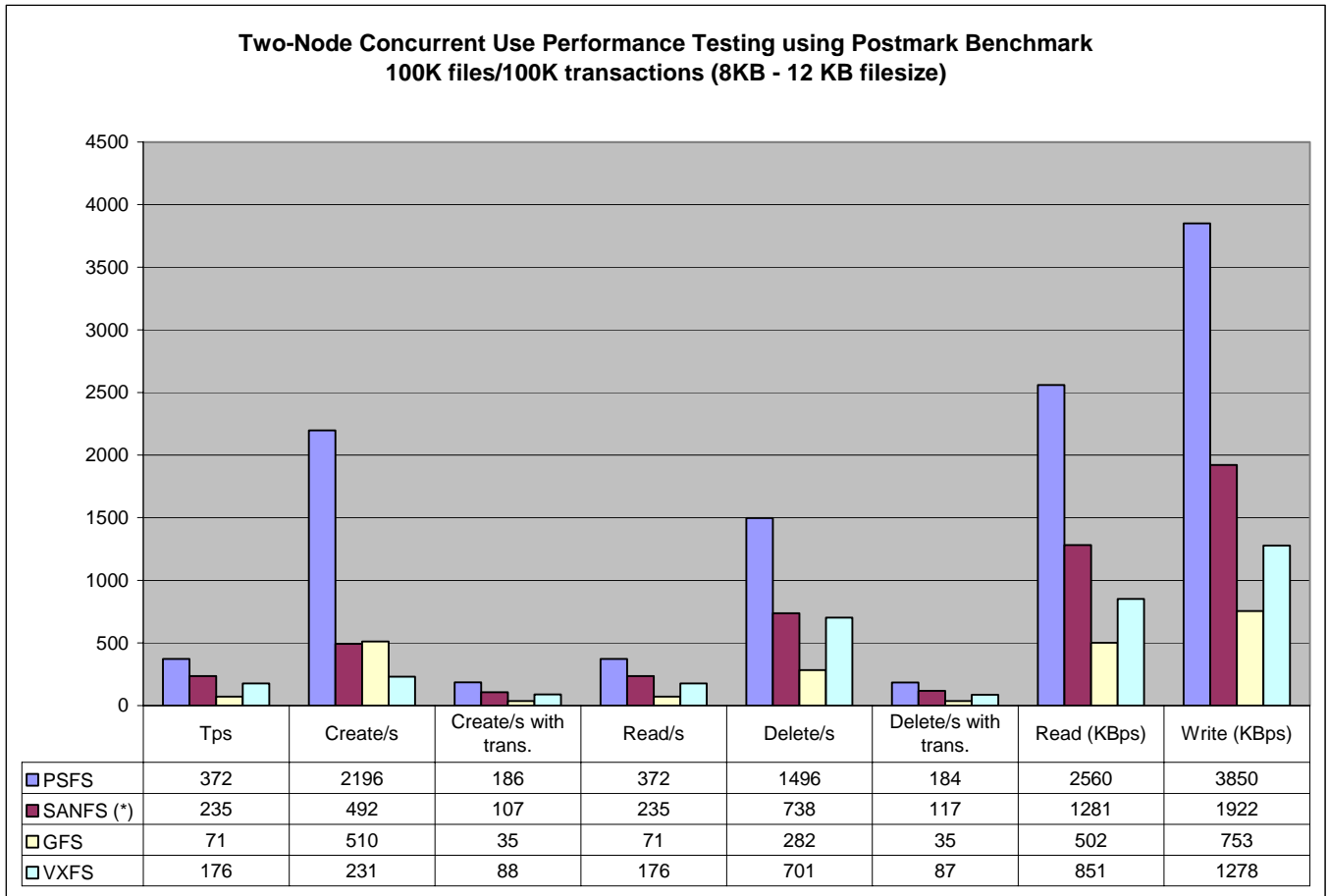
For SANFS, (\*) highlights it was used a different SAN configuration.



Second workload:

A single Postmark process ran on each of the two servers. Each server processed 100K files (8KB-12KB file size) with 100K transactions.

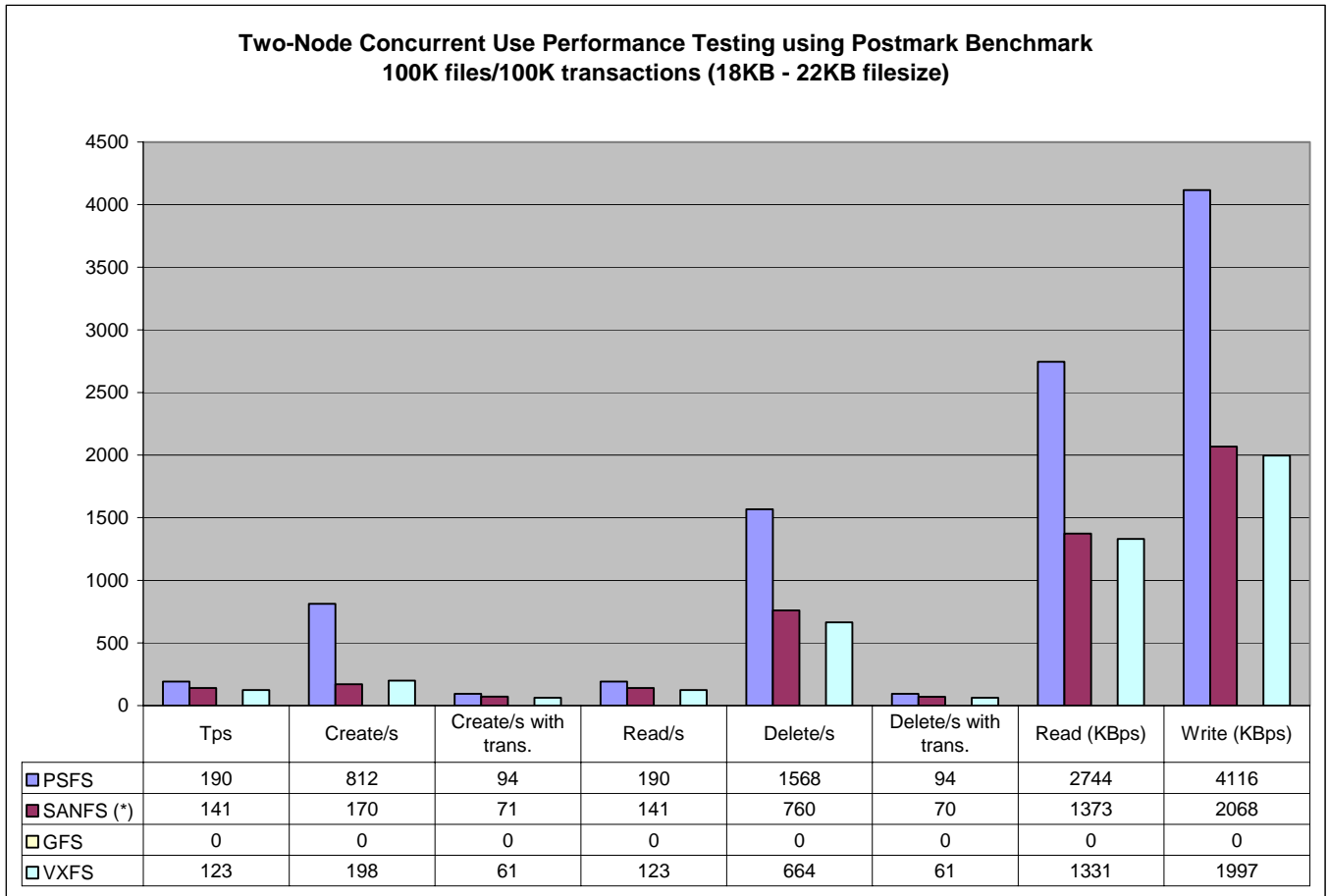
For SANFS, (\*) highlights it was used a different SAN configuration.



Third workload:

A single Postmark process ran on each of the two servers. Each server processed 100K files (18KB-22KB file size) with 100K transactions.

For SANFS, (\*) highlights it was used a different SAN configuration.



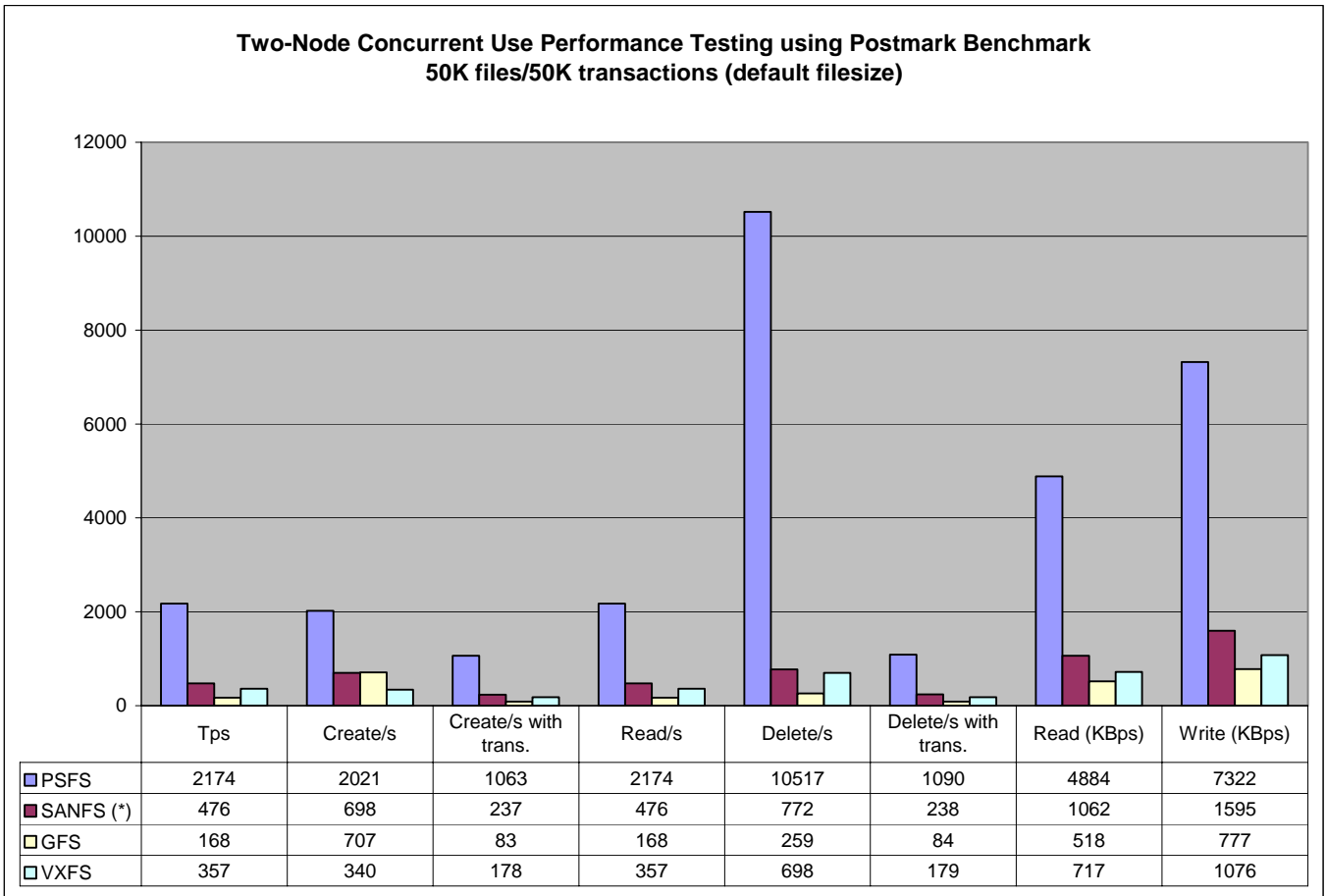
Note: Red Hat GFS was unable to finish this test and therefore showed no results.

## 5.5. Second dataset

### First workload:

A single Postmark process ran on each of the two servers. Each server processed 50K files of default file size (0.5KB-10KB) with 50K transactions.

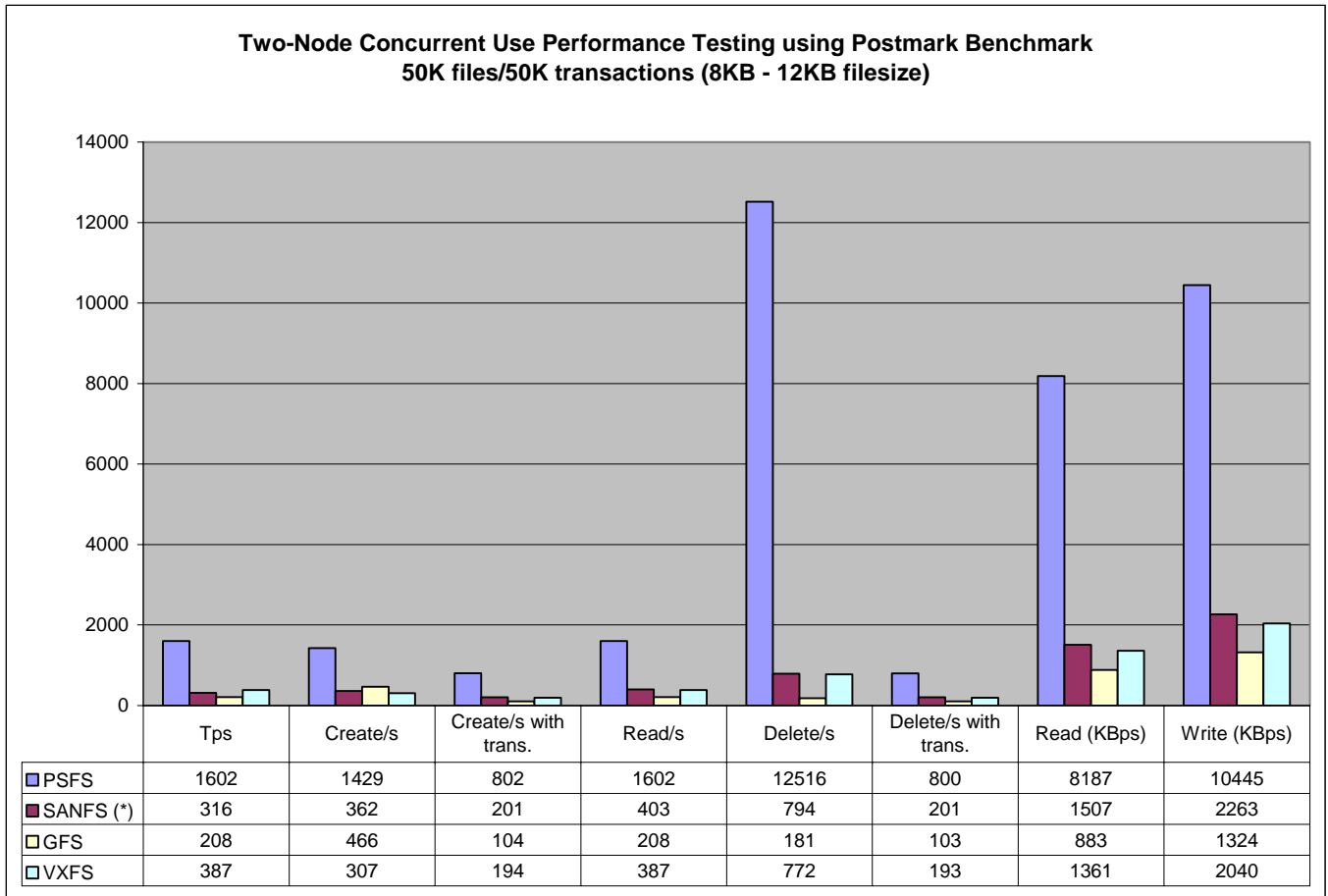
For SANFS, (\*) highlights it was used a different SAN configuration.



Second workload:

A single Postmark process ran on each of the two servers. Each server processed 50K files (8KB-12KB file size) with 50K transactions.

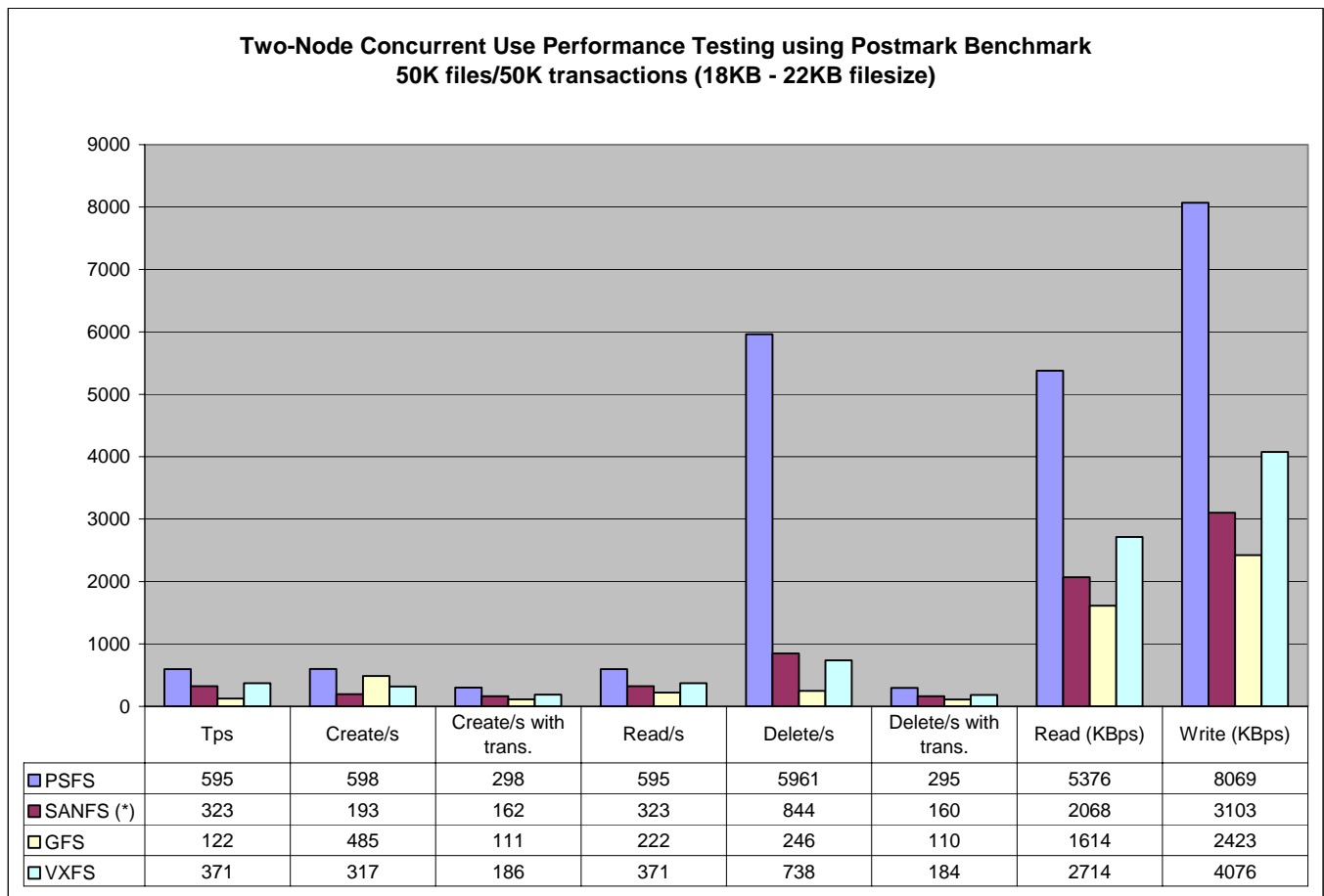
For SANFS, (\*) highlights it was used a different SAN configuration.



### Third workload:

A single Postmark process ran on each of the two servers. Each server processed 100K files (18KB-22KB file size) with 100K transactions.

For SANFS, (\*) highlights it was used a different SAN configuration.



## 6. Third test

### 6.1. Goal

Evaluate the performance of the Linux cluster file systems for a large scale e-mail service, when an IMAP server has to “stat” messages stored in a Maildir format. Maildir stores each message in a single file in three directories per user.

### 6.2. The benchmark

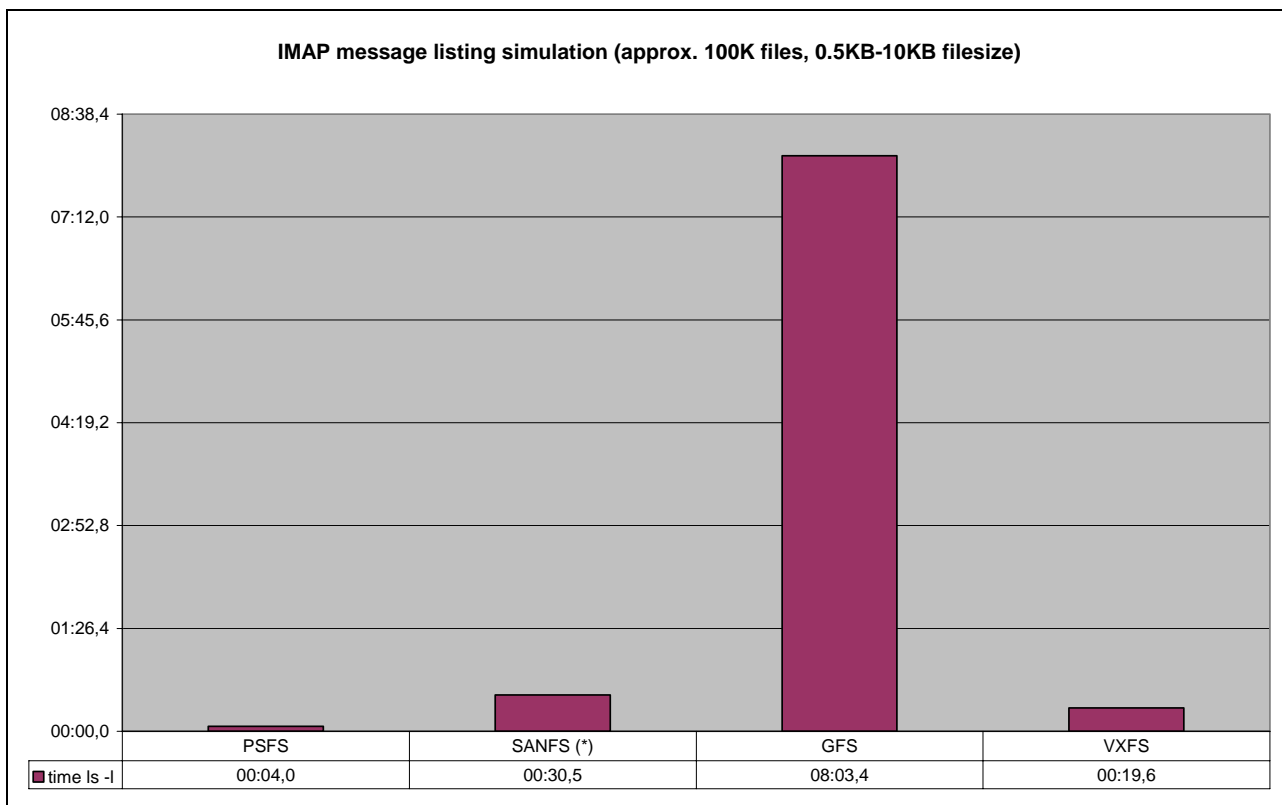
We simulated an IMAP listing of a user’s message folder by performing a “`ls -l * > /dev/null`” in a directory containing approximately 100K files (0.5-10K file size). The results are in seconds. (Less is better, time is in seconds).

For SANFS, (\*) highlights it was used a different SAN configuration.

### 6.3. Players

PSFS: Polyserve Matrix Server 2.6.1  
SANFS: IBM TotalStorage San File System 2.1  
GFS: Red Hat GFS 6.0  
VXFS: Veritas Cluster File System 4.0

### 6.4. The Dataset



## 6. Conclusions

Cluster file systems are now a mature technology and the market today shows a variety of solutions. Each one addresses a particular need and represents a specific vision of data access concurrency, and therefore, they show differences in performance. For instance, IBM TotalStorage San File System, has the capability to share data between server platforms, while Veritas Cluster File System incorporates all well known storage management features of the Veritas Volume Manager.

The test we performed highlighted how Polyserve Matrix Server is focused on file serving, taking advantage of its symmetric design and reduced metadata traffic to achieve outstanding performance in small file concurrent access, which – as we have said – is a very challenging workload for cluster file system architectures.