VERITAS®

VERITAS® File Server Edition™ Performance Brief: A PostMark 1.11 Benchmark Comparison

1. Introduction

This brief documents the performance of release 5.0 of the VERITAS File Server Edition when used with Sun Microsystems' Solaris 7 operating system. Testing was done to compare the VERITAS File Server Edition and the UNIX File System (UFS) which is included with the Solaris 7 operating system. The test suite was run on both the 64-bit and 32-bit versions of Solaris 7, to determine if there were differences in performance between these two versions. The VERITAS products showed performance improvements of up to 13 times over comparable UFS configurations. This paper also describes the standardized test that was used to evaluate the performance as well as the environment under which the performance improvements were observed.

The VERITAS File Server Edition is an integrated set of four VERITAS products, configuration guidelines, and the Samba CIFS Package. The VERITAS File Server Edition includes the following packages:

- VERITAS File System (VxFS®)
- VERITAS Volume Manager (VxVM®)
- VERITAS Storage Administrator (VMSA)
- VERITAS QuickLog (VxQLOG)
- Samba

These products provide for significant performance increases over the default Solaris file system, superior ease of use, as well as connectivity for PCs.

The volume used for testing was configured with VxVM in an eight-way striped (RAID-0) configuration. During testing one file system, either VxFS or UFS was set up on this volume. This intentionally limited configuration "de-tunes" the computer system and forces the benchmark to measure the performance of the file systems and not the maximum performance of the computer system as a whole.

To properly evaluate the performance of the various file systems a standardized benchmark must be used. In the tests for this paper we used Network Appliance's PostMark benchmark, version 1.11. A benchmark of this type is far better suited to performance testing than other extemporaneous "benchmarks" such as the UNIX `dd` command. Using such commands as benchmarks can lead to false conclusions when environmental factors such as client caching lead to irrelevant "throughput" results.

2. About PostMark 1.11

(The italicized portion of this section contains excerpts from the Network Appliance's TR3022 "PostMark: A New File System Benchmark" available at http://www.netapp.com/technology/level3/3022.html).

Abstract

Existing file system benchmarks are deficient in portraying performance in the ephemeral small-file regime used by Internet software, especially:

- *electronic mail;*
- *netnews; and*
- *web-based commerce.*

PostMark is a new benchmark to measure performance for this class of application.

Introduction

Although a panoply of file system benchmarks currently exists, most emphasize raw throughput or performance on large, relatively long-lived groups of files. Current file servers provide services such as electronic mail, netnews, and commerce service, which depend on enormous numbers of relatively short- lived files. These systems are inherited from an era predating the exponential growth of the Internet and were never envisioned as being scaleable to today's required levels.

When planning the acquisition and growth of networked storage systems, it is difficult to predict the performance for Internet duties from existing benchmarks. The only recourse is to measure the performance under the load of electronic mail and netnews services. As using production systems for performance evaluation is both resource intensive and non-deterministic (the load is varied and seldom reproducible), PostMark was created to simulate heavy small-file system loads with a minimal amount of software and configuration effort and to provide complete reproducibility.

The Benchmark Software

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.

Appending data to a file opens a random file, seeks to its current end, and writes a random amount of data. This value is chosen to be less than the configured file size high bound. If the file is already at the maximum size, no further data will appended.

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

On completion of each run, a report is generated showing:

- *Elapsed time*
- *Elapsed time spent performing transactions and average transaction rate (files/second)*
- *Total number of files created and average creation rate (files/second)*
- *Number of files created initially and average creation rate (files/second)*
- *Number of files created during sequence of transactions and average creation rate (files/second)*
- *Total number of files read and average rate (files/second)*
- *Total number of files appended and average rate (files/second)*
- *Total number of files deleted and average deletion rate (files/second)*
- *Number of files deleted after transactions were complete and average deletion rate (files/second)*
- *Number of files deleted during sequence of transactions and average deletion rate (files/second)*
- *Total size of data read and average input rate (bytes/second)*
- *Total size of data written and average output rate (bytes/second)*

A portable random number generator (derived from the Unix reference implementation) is included in the PostMark distribution ensuring identical initial conditions across different platforms and operating systems.

3. Tested Configuration

Tests were run on a Sun Microsystems Ultra Enterprise 4000 system with four 167 MHz UltraSparc processors and 832 MB of memory. The file system under test was configured on disks contained in two Sun Microsystems D1000 storage subsystems. The storage subsystem drives used were 7200 RPM, 9 GB Seagate Barracudas ST39173W attached to the Ultra system via four fast wide differential SCSI Sbus controllers. Each SCSI bus had six drives attached to it, of which two were used.

In this paper we are attempting to characterize the performance of one portion of a computer system (the file system portion) using the PostMark 1.11 benchmark. We have therefore configured the system to use only one file system distributed across eight disks, rather than a large pool of file systems and disks. The computer system was configured with ample CPU and memory resources. This is intended to force the benchmark to bottleneck on the file system, and therefore reflect file system performance.

The following software releases were used in testing:

- Solaris 7
- VxFS 3.3.2
- VxVM 3.0.1
- VxQLOG 1.1

The file system under test was configured over eight disks, under the control of VxVM. One 25 GB striped (RAID–0) volume was used during testing.

All the *vxtunefs* parameters were set at default as follows:

- read_pref_io=65536
- read_nstream=8
- read_unit_io=65536
- write_pref_io=65536
- write_nstream=8
- write_unit_io=65536
- pref_strength=20
- buf_breakup_size=262144
- discovered_direct_iosz=262144
- max_direct_iosz=8388608
- default_indr_size=8192
- qio_cache_enable=0
- max_diskq=3145728
- initial_extent=1
- max_seqio_extent_size=2048

The mkfs parameters for VxFS chosen were:

- bsize=1024
- logsize=16384

A 2 GB volume was configured on a separate single disk that was only used for the QuickLog log.

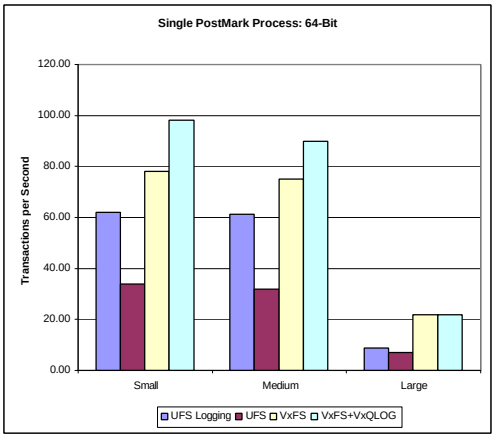
4. Test Results

In this paper we have chosen to show the average transaction rate (files/second). We felt that this is the most relevant metric to gauge computer systems being utilized for electronic mail, etc. We have also left all the PostMark benchmark parameters at their default settings.

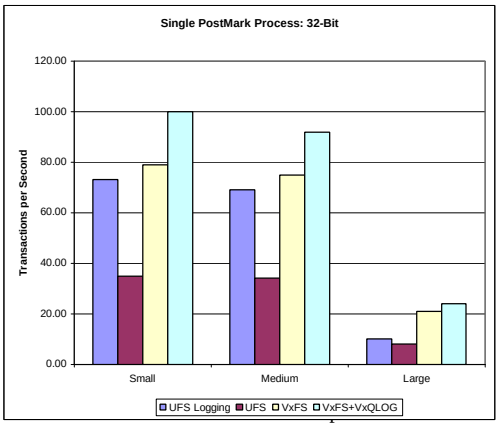
The results of the testing are shown in the tables and charts below. In these charts of PostMark results, "goodness" is up.

4.1 Single PostMark Process

64 Bit													
Series Name	Test Parameters				Average Transactions per second				Percent Improvement of UFS-L over UFS	Percent Improvement of VxFS over UFS	Percent Improvement of VxFS over UFS-L	Percent Improvement of VxFS+VxQLOG over UFS	Percent Improvement of VxFS+VxQLOG over UFS-L
	# PostMark Procs	# Sub Dir	# Files	# Trans	UFS Logging	UFS	VxFS	VxFS+ VxQLOG					
Small	1	1	20,000	50,000	62.00	34.00	78.00	98.00	82.35	129.41	25.81	188.24	58.06
Medium	1	1	20,000	100,000	61.00	32.00	75.00	90.00	90.63	134.38	22.95	181.25	47.54
Large	1	1	200,000	50,000	9.00	7.00	22.00	22.00	28.57	214.29	144.44	214.29	144.44



32 Bit													
Series Name	Test Parameters				Average Transactions per second				Percent Improvement of UFS-L over UFS	Percent Improvement of VxFS over UFS	Percent Improvement of VxFS over UFS-L	Percent Improvement of VxFS+VxQLOG over UFS	Percent Improvement of VxFS+VxQLOG over UFS-L
	# PostMark Procs	# Sub Dir	# Files	# Trans	UFS Logging	UFS	VxFS	VxFS+ VxQLOG					
Small	1	1	20,000	50,000	73.00	35.00	79.00	100.00	108.57	125.71	8.22	185.71	36.99
Medium	1	1	20,000	100,000	69.00	34.00	75.00	92.00	102.94	120.59	8.70	170.59	33.33
Large	1	1	200,000	50,000	10.00	8.00	21.00	24.00	25.00	162.50	110.00	200.00	140.00



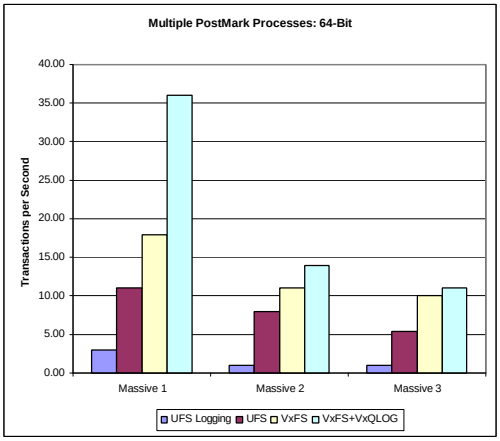
From these results, we see that VxFS, although a "logging" file system, exceeds the throughput of the standard UFS file system by a maximum of 214%. Comparing VxFS to Solaris 7's logging option, we see that VxFS exceeds the throughput of UFS by a maximum of 144%.

We also see that VxFS+VxQLOG exceeds the throughput of the standard UFS file system by a maximum of 214%. Comparing VxFS+VxQLOG to Solaris 7's logging option, we see that VxFS+VxQLOG exceeds the throughput of UFS by a maximum of 144%.

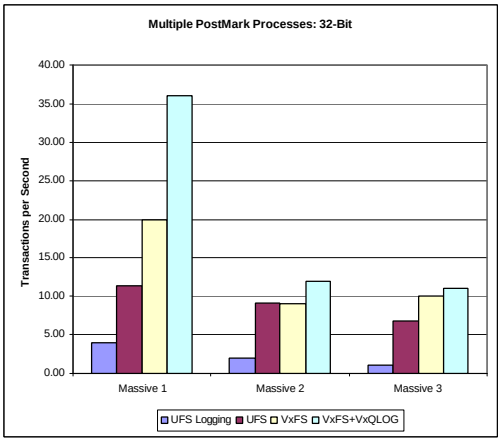
4.2 Multiple PostMark Processes

The preceding series of tests have one flaw, they use only one process to do all the transactions. Multiple processes are more representative of the real world and shows the scalability of the system under test. What would happen if we used more than one process? (Note: the Sub Dir, Files, Trans parameters, and the Average Transactions per second results are on a per process basis.)

64 Bit													
	Test Parameters				Average Transactions per second								
Series Name	# PostMark Procs	# Sub Dir	# Files	# Trans	UFS Logging	UFS	VxFS	VxFS+ VxQLOG	Percent Improvement of UFS-L over UFS	Percent Improvement of VxFS over UFS	Percent Improvement of VxFS over UFS-L	Percent Improvement of VxFS+VxQLOG over UFS	Percent Improvement of VxFS+VxQLOG over UFS-L
Massive 1	12	100	2,000	50,000	3.00	11.08	18.00	36.00	-72.92	62.45	500.00	224.91	1,100.00
Massive 2	12	100	20,000	50,000	1.00	8.00	11.00	14.00	-87.50	37.50	1,000.00	75.00	1,300.00
Massive 3	12	100	200,000	50,000	1.00	5.42	10.00	11.00	-81.55	84.50	900.00	102.95	1,000.00



32 Bit													
Series Name	Test Parameters				Average Transactions per second				Percent Improvement of UFS-L over UFS	Percent Improvement of VxFS over UFS	Percent Improvement of VxFS over UFS-L	Percent Improvement of VxFS+VxQLOG over UFS	Percent Improvement of VxFS+VxQLOG over UFS-L
	# PostMark Procs	# Sub Dir	# Files	# Trans	UFS Logging	UFS	VxFS	VxFS+ VxQLOG					
Massive 1	12	100	2,000	50,000	4.00	11.33	20.00	36.00	-64.70	76.52	400.00	217.74	800.00
Massive 2	12	100	20,000	50,000	1.92	9.08	9.00	12.00	-78.85	-0.88	368.75	32.16	525.00
Massive 3	12	100	200,000	50,000	1.00	6.83	10.00	11.00	-85.36	46.41	900.00	61.05	1,000.00



From these results, we see that VxFS, although a "logging" file system, exceeds the throughput of the standard UFS file system by a maximum of 85%. Comparing VxFS to Solaris 7's logging option, we see that VxFS exceeds UFS by a maximum of 1,000%.

We also see that VxFS+VxQLOG exceeds the throughput of the standard UFS file system by a maximum of 225%. Comparing VxFS+VxQLOG to Solaris 7's logging option, we see that VxFS+VxQLOG exceeds the throughput of UFS by a maximum of 1,300%.

5. Analysis of Results

In this study of VERITAS Software's File Server Edition as compared to Solaris 7 performance we see that VxFS is a maximum of 1,000% improvement over Solaris 7's logging option and a maximum of 214% improvement over the default Solaris 7 no-logging. It must be remembered that VxFS is a "logging" type of File System and that it shows a moderate performance improvement over Solaris 7's default no-logging option and an even better performance improvement over Solaris 7's logging option.

Improvement of Throughput Performance of VxFS Compared to UFS			
		64-bit	32-bit
No-logging	1 Proc	129-214%	121-163%
	12 Proc	38-85%	-1-77%
Logging	1 Proc	23-144%	8-110%
	12 Proc	500-1,000%	369-900%

Also in this study we see that VxFS+VxQLOG is a maximum of 1,300% improvement over Solaris 7's logging option and a maximum of 225% improvement over the default Solaris 7 no-logging.

Improvement of Throughput Performance of VxFS+VxQLOG Compared to UFS			
		64-bit	32-bit
No-logging	1 Proc	181-214%	171-200%
	12 Proc	75-225%	32-218%
Logging	1 Proc	48-144%	33-140%
	12 Proc	1,000-1,300%	525-1,000%

It is also interesting to note the performance of Solaris 7's default no-logging option as compared to the performance of Solaris 7's logging. Note that UFS's logging option can **degrade** performance.

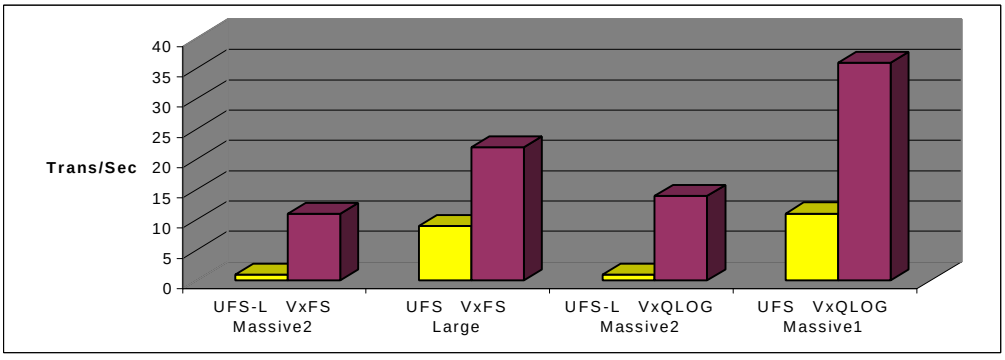
Improvement of Throughput Performance of UFS No-logging Compared to UFS Logging		
	64-bit	32-bit
1 Proc	29-91%	25-109%
12 Proc	-88-(-77)%	-85-(-65)%

6. Summary

The PostMark 1.11 benchmark is generally used to evaluate the performance of an optimally configured and tuned configuration of a specific computer system. It reflects typical electronic mail, netnews, and web-based commerce workloads, though specific sites' operation mixes may differ somewhat.

By "de-tuning" the system configuration, we can show the effect of one specific system component, the file system, on performance. In addition to performance, the ability of the server to recover quickly from failures to minimize service reduction should be considered.

VxFS is an improvement of a maximum of 11 times over the similarly functional UFS logging and an improvement of a maximum of three times over the default UFS no-logging, while VxFS+VxQLOG is an improvement of a maximum of 14 times over UFS logging and a maximum of a three times improvement over UFS no-logging.



VERITAS Software Corporation
1600 Plymouth Street Mountain View, CA 94043
(800) 258-8649 (415) 335-8000 Fax: (415) 335-8050
vx-sales@veritas.com http://www.veritas.com/