White Paper

File System Throughput Performance on RedHawk™ Linux

By: Nikhil Nanal
Concurrent Real-Time
August 2016

Introduction

This paper reports the throughput performance of the EXT4, XFS, BTRFS and REISER4 file systems on RedHawk Linux 7.2 running on an Intel Xeon server. Measurements were taken using the IOZone file system bench marking tool using a 480 GB solid state disk and a 6 TB spinning hard disk.

The study shows that BTRFS gives the best write throughput for small file sizes, while XFS and EXT4 file systems have better read throughput. REISER4 gave the best overall performance for large file sizes.

Overview

A file system plays a central role in determining the overall performance of an operating system. It serves as the interface for interacting with external devices such as the disk. Accessing storage devices may introduce significant latency when making the data available to an application thus affecting system performance. To reduce this latency, file system implementations have incorporated features to optimize when and how data is read from or written to the disk.

The Linux kernel supports a number of block-based file systems like the EXT2/3/4, XFS, BTRFS, etc. This flexibility of choice makes it important to evaluate which file system delivers the best throughput performance for specific application goals. The goal of this study is to compare the performance of different file systems under similar workloads and system configurations and identify the file system that would best suit the end user's requirements.

Section 3 of this report describes the configuration of the system under test. Section 4 gives a brief overview of the IOZone benchmarking tool that was used for this study along with the various options enabled to create conditions that are likely to affect the throughput performance and provide realistic performance results. Section 5 describes the methodology which was followed to perform the test. Section 6 contains observations and results followed by concluding remarks in Section 7.

System Configuration

- Operating System: RedHawk Linux 7.2 x86_64
- Processor Model: Dual 2.3 GHz Intel® Xeon® E5-2670 v3
- Processor Cache Size: 30 MB
- Processor Cores: 12 cores per CPU
- Main Memory: 32 GB
- Disks Tested:
  - 480 GB SSD, SATA III
  - 6 TB hard drive, SATA III

- File systems under test: EXT4, XFS, BTRFS, REISER4
- Benchmark: IOZone 3.424 compiled for 64-bit mode.

IOZone Benchmark

IOZone is a general purpose file system benchmarking too that provides multiple command line options that can be set to define the scope of a test. A complete list of parameters can be found in the IOZone documentation. The following options were set in the tests performed in this study.

- Record Length (-r): Specifies the size of each I/O request. The number of operations multiplied by the record size equals the total file size.
- Minimum File Size (-m): The minimum file size for which the test is carried out should be equal to the record length. In this case the number of operations = 1.
- Maximum File Size (-g): The maximum file size can be any arbitrary value. It is recommended that it be set to at least twice the total system memory size, so that memory and cache buffer effects can be transcended and the effect of disk access on the throughput can be observed. File sizes are specified with units of K, M or G for kilobytes, megabytes or gigabytes respectively.
- Processor Cache Line Size (-S): Defines the processor cache size (in Kbytes). It is used by the IOZone code internally for buffer alignment and purge functionality.
- Test Number (-i): IOZone measures throughput for sequential writes, sequential reads, random read, random write, re-read, re-write etc. A complete list can be found in the IOZone documentation.
- Cache Purge enable(-p): Purges processor cache.

Test Procedure

The following steps were taken to perform the file system benchmarking.

1. Create a partition
2. Build a Linux file system
3. Mount the file system
4. Install the IOZone benchmark rpm or build and install its custom package
5. Execute the IOZone test, taking care to flush file system buffers and drop caches before the run of each test.
6. Plot the test results using gnuplot.

Observations and Discussion of Results

Throughput measurements were made for the following six operations: Write sequential, Read sequential, Re-read, Re-write, Random read and Random write. The bullets below summarize the results of the tests.

Noticeable is the performance of the file systems in two separate regions -- the Low File Size region where the file size is still less than the system memory size (up to 4 GB) and in the High File Size region where file size is more than memory size (32 GB to 128 GB). The performance of the file systems is seen to vary under both conditions. These observations apply only to the file systems tested in this study.

- **480 GB SSD Performance:**
  - **SEQUENTIALWRITE:**
    - BTRFS gives the highest write throughput for small file sizes.
    - As the file size increases, REISER4 performs much better.
    - EXT4 is the worst performer, especially for smaller files.
  - **SEQUENTIAL READ:**
    - EXT4 and XFS have the best read throughput. BTRFS and REISER4 follow closely behind.
    - With larger files, EXT4, XFS and BTRFS show similar performance, though REISER4 performed slightly better.
  - **RE-READ:**
    - XFS and EXT4 give the best re-read performance for small sizes, whereas REISER4 is the best performer for larger files, followed closely by BTRFS.
  - **RE-WRITE:**
    - XFS and BTRFS are the best performers with small files and REISER4 with larger files. EXT4 is the slowest.
  - **RANDOM READ/WRITE:**
    - REISER4 shows poorest but still respectable performance for small files, however it maintains a higher throughput with larger files, similar to all other tests.
    - BTRFS outperforms all other file systems with XFS following closely.

- **6 TB HDD Performance:**
  - **SEQUENTIAL WRITE:**
    - BTRFS is the best performer for small files, followed closely by REISER4.
    - With larger files, REISER4 is significantly better than all other file systems.
    - EXT4 has the poorest write performance among all four file systems.
  - **SEQUENTIAL READ/RE-READ:**
    - All file systems have similar performance, except with large files where REISER4 is the best.
  - **RE-WRITE:**
    - BTRFS gives the best performance for small files.
    - XFS and REISER4 compete closely for second best for small files.
    - REISER4 performs significantly better with larger files.
  - **RANDOM WRITE:**
    - BTRFS is the best performer with small files and REISER4 with larger files. XFS follows BTRFS closely.
    - REISER4 is not as good as XFS in random seeks. EXT4 gives the poorest random write throughput performance.
  - **RANDOM READ:**
    - Similar to sequential read, random read performance of all file systems is found to be nearly the same.

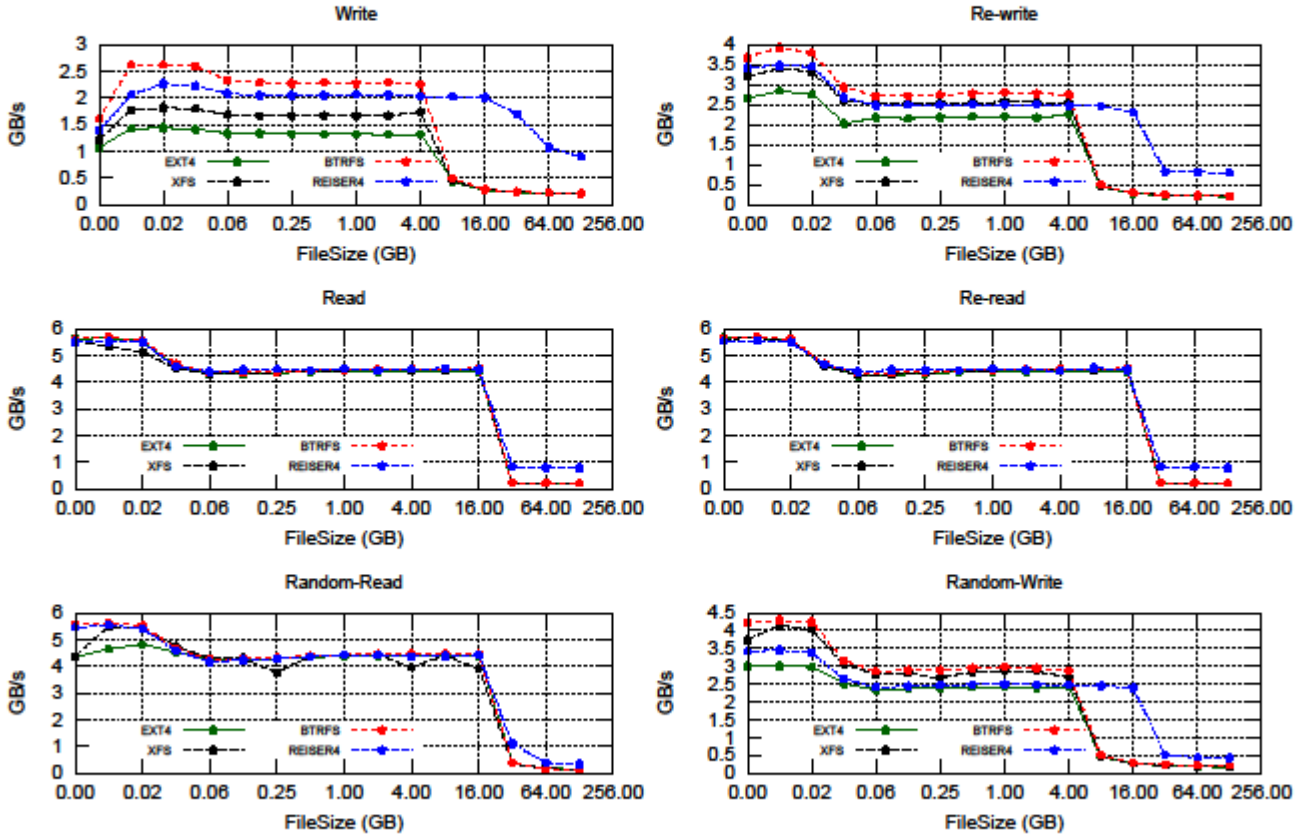
FileSystem Performance (Throughput) Comparsion for 6TB HDD Using IOZone.

System:Intel(R) Xeon(R)CPU E5-2670 v3, 2.30GHz,32GB RAM, 12 Core HTT

Maximum file size of 128 Gigabytes.Minimum file size of 4 Megabytes.Record Size 4 Megabytes Machine = Linux beast 4.1.15-rt17-RedHawk-7.2-custom #1 SMP PREEMPT Wed Jun 22 Purge Mode On

Output is in GBytes/sec Processor cache size set to 30720 kBytes. Processor cache line size set to 64 bytes.File stride size set to 17 * record size.

Command line used: iozone -a -g 128G -r 4m -n 4m -S 30720 -L 64 -M -p -I0 [-I1 -I2]



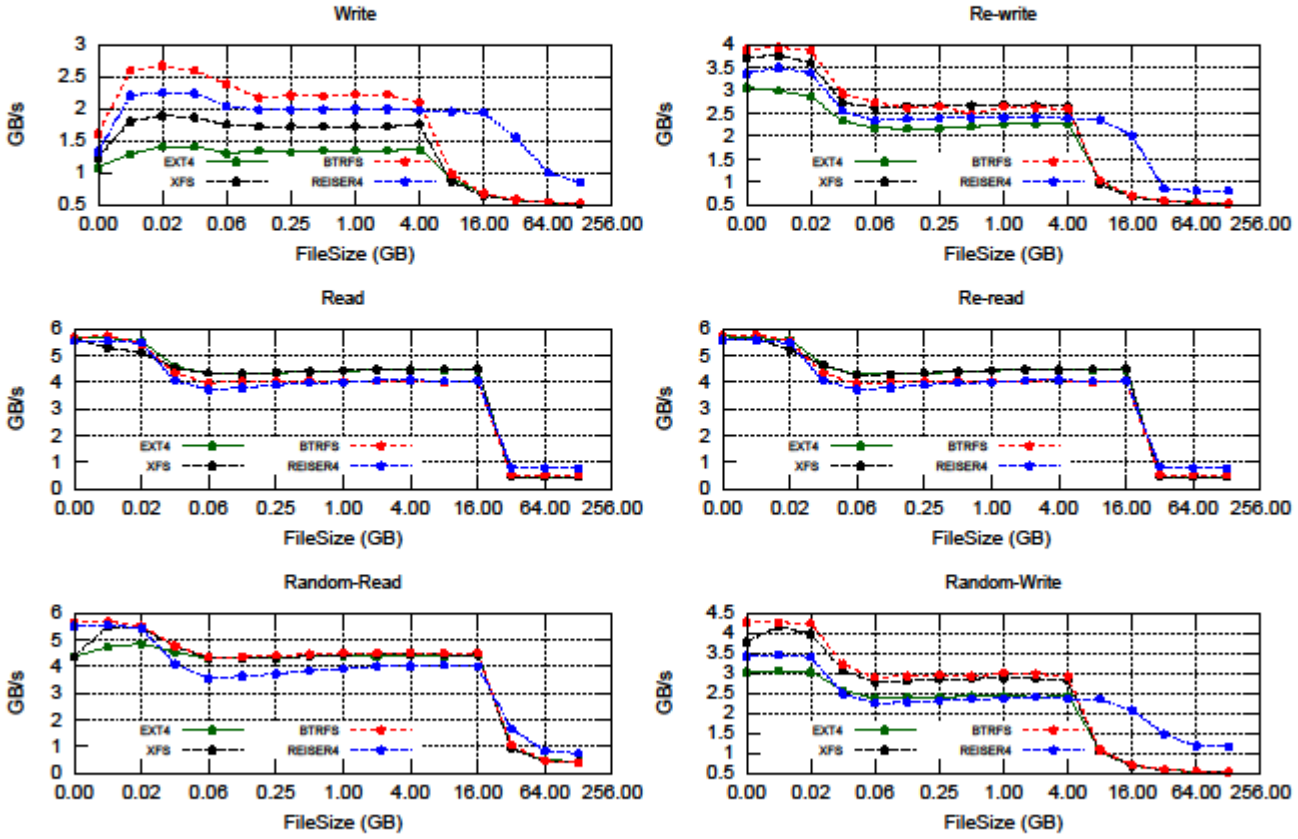
FileSystem Performance (Throughput) Comparsion for 480 GB SSD Using IOZone.

System:Intel(R) Xeon(R)CPU E5-2670 v3, 2.30GHz,32GB RAM, 12 Core HTT

Maximum file size of 128 Gigabytes.Minimum file size of 4 Megabytes.Record Size 4 Megabytes Machine = Linux beast 4.1.15-rt17-RedHawk-7.2-custom #1 SMP PREEMPT Wed Jun 22 Purge Mode On

Output is in GBytes/sec Processor cache size set to 30720 kBytes. Processor cache line size set to 64 bytes.File stride size set to 17 * record size.

Command line used: iozone -a -g 128G -r 4m -n 4m -S 30720 -L 64 -M -p -I0 [-I1 -I2]



Conclusion

The RedHawk 7.2 kernel mounted with BTRFS is a good option for applications that are write-intensive whereas XFS is a good file system for applications which are read intensive. For general purpose applications, BTRFS could be a good choice for both reads and writes. However, this is true only for small files. If an application uses larger files most of the time, then the REISER4 file system shows significantly higher performance in reads and writes. For random, burst applications, XFS is recommended regardless of the file size as it gives more consistent throughput.

It should be noted that these recommendations are based only on this throughput study. Other application needs such as crash resiliency and data integrity may affect throughput to different extents and warrant the selection of a different file system