Topic 14: File System Performance Characterization

Ext4 vs. XFS vs. Btrfs

Presenters: Aobo GUO 225040143, Jianqiu WU 225040310

School of Data Science / The Chinese University of Hong Kong, Shenzhen

April 23, 2026

Goal and Evaluation Scope

- **Goal:** compare **Ext4**, **XFS**, and **Btrfs** under representative Linux storage workloads.
- **Tools:** `fio`, `dd`, and `sysbench fileio`.
- **Workloads:**
  - **Baseline:** sequential / random read and write
  - **Video Editor:** large concurrent sequential I/O
  - **Mail Server:** many small files with buffered sync-heavy writes
- **Metrics:** throughput, IOPS, latency percentiles, and CPU usage *(user/system when available)*.
- **Tuning:** `noatime`, `data=writeback`, `barrier=0`, and attempted XFS `nobarrier`.

Division of Work

Aobo GUO: baseline `fio` comparison, IOPS analysis, and latency analysis	(50%)
Jianqiu WU: `dd` validation, workload evaluation, and tuning analysis	(50%)

Experimental Setup and Interpretation Rules

Item	Value
OS	Linux 6.8.0-107-generic
CPU	4 cores
Memory	8.1 *GB*
Filesystems	Ext4 / XFS / Btrfs
Runs	FIO repeated 3 times
Latency stats	Avg / P50 / P95 / P99
CPU stats	User/System (`fio`); total CPU% (`sysbench`)

Workload Settings

Workload	Key settings
Baseline FIO	30*s*; `iodepth=16`; `direct=1`
Video Editor	4 jobs; 1 *MB*; `iodepth=32`; 45*s*
Mail Server	10,000 files; 4 *KB*; 4 threads; 45*s*
`dd` check	1 *GB* direct sequential read/write

Interpretation Rule

`fio`/`dd` use direct I/O, while Mail Server `sysbench` uses buffered I/O with frequent `fsync()`. These results answer different questions and should not be ranked as a single leaderboard.

Very Brief Intuition Before the Experiments

- **Ext4:** mature and usually low overhead, so it may stay strong on small random writes.
- **XFS:** designed for scalable allocation, so it may benefit large sequential and concurrent I/O.
- **Btrfs:** richer semantics through CoW and checksums, but random or metadata-heavy writes may cost more.

Main Expectation

There is **no universal winner**. The best filesystem depends on workload pattern, queue depth, and persistence semantics.

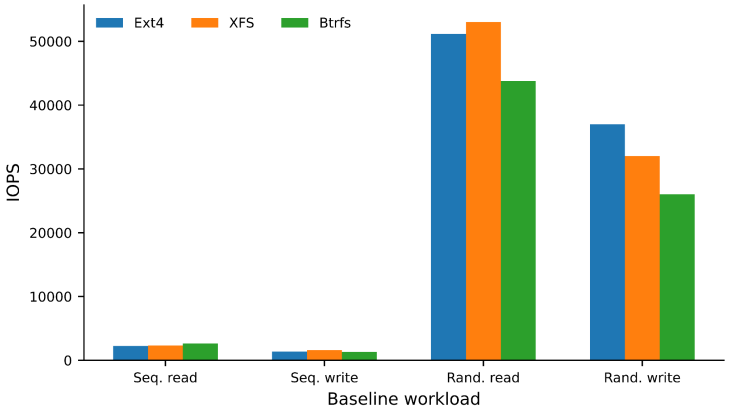
Baseline FIO: The Winner Changes with Access Pattern

Workload	Winner	Evidence *MB/s or IOPS; P50 ms*	Main message
Seq. read	Btrfs	2626, 5.78	All three read well sequentially; Btrfs tops this run.
Seq. write	XFS	1589, 9.85	XFS shows the clearest streaming-write advantage.
Rand. read	XFS	53.0k, 0.232	XFS serves 4 KB random reads most efficiently.
Rand. write	Ext4	37.0k, 0.327	Ext4 gives the best small-write result here.

Takeaway

- **Performance:** **Ext4** leads direct random write, while **XFS** has the strongest sequential-write profile.
- **CPU note:** random FIO jobs are system-CPU-heavy (Ext4/XFS ∼72–78%), while **Btrfs** shows the highest system CPU on sequential write (31.13%).

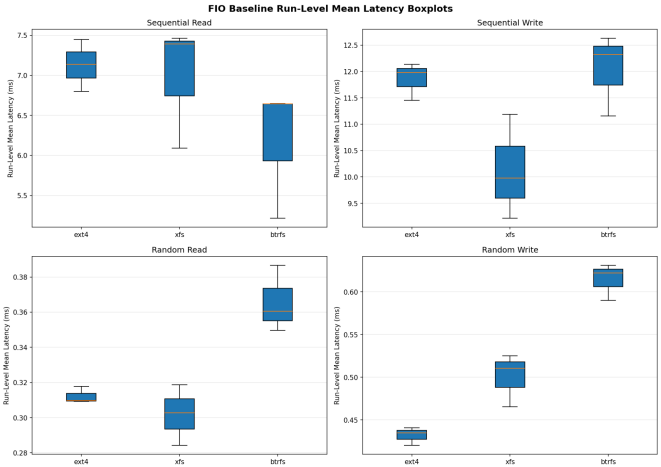
Baseline IOPS: Access Pattern Changes the Leader



Key Observations

- Sequential jobs emphasize throughput.
- Random jobs expose request service rate.
- XFS is strongest on seq. write and rand. read.
- Ext4 leads rand. write.

Baseline Latency Boxplot: Run-Level Mean Latency Distribution



Key Observations

- **Ext4:** lowest latency on rand. write.
- **XFS:** best on seq. write and rand. read.
- **Btrfs:** good on seq. read, but costlier on rand. write.

Sanity Check with dd: Direct Sequential I/O Cross-Check

FS	Seq. write *GB/s*	Seq. read *GB/s*	Write CPU *%*	Read CPU *%*
Ext4	1.10	1.70	11	9
XFS	2.10	1.50	20	8
Btrfs	2.20	1.30	29	6

Key Observations

- **Role:** dd is a direct-I/O sanity check for large-block, single-thread sequential transfer.
- **Result:** all three filesystems sustain >1 *GB/s*; XFS/Btrfs write faster here, while Ext4 reads fastest in this check.
- **Caveat:** dd is synchronous and short-lived, so its numbers may differ from FIO; quantitative ranking should still rely on FIO.

Video Editor Workload: Strong Streaming Throughput on All Three

FS	Seq. write *MB/s*	Seq. read *MB/s*	Write P50 *ms*	Read P50 *ms*
Ext4	1196.9	2070.2	103.63	57.23
XFS	1239.3	1545.5	97.34	74.62
Btrfs	1243.5	1527.3	96.64	83.62

Key Observations

- **Overall:** all three reach GB/s-class streaming throughput.
- **Write side:** XFS and Btrfs are slightly ahead of Ext4.
- **Read side:** Ext4 performs best in this run.
- **Trade-off:** deeper queues improve throughput but increase latency; user CPU stays low, but **Btrfs** sequential write uses much higher system CPU (12.35%) than Ext4/XFS (2.29–2.69%).

Mail Server Workload: Ext4 and XFS Are the Stable Choices

Setup

Tool	`sysbench fileio`	**Files**	10,000 small files
Access	random read/write	**Threads**	4
Duration	45 *s*	**Fsync**	`-file-fsync-freq=1`

Note: scaled down to fit current VM disk/inode limits; it still targets metadata persistence and journaling behavior.

FS	Events */s*	Fsyncs */s*	Avg. lat. *(ms)*	Max lat. *(ms)*	CPU *(%)*
Ext4	16.4k	17.2k	0.23	24.35	8
XFS	16.1k	16.9k	0.24	23.33	8
Btrfs	2.24M	2.24M	0.00	306.13	219

Interpretation

- **Ext4/XFS:** stable and interpretable, with similar `events/s`, low latency, and modest CPU use (Ext4: 1.00s user + 2.93s system; XFS: 1.07s + 2.70s).
- **Btrfs:** not a true "win": `events/s` is a composite metric dominated here by abnormal `fsyncs/s`; avg./P95 collapse to 0.00 due to output precision, max latency reaches 306 ms, and CPU surges to 41.06s user + 57.69s system (219%).

Tuning Results: Gains Are Limited and Safety Matters

Common Tuning: `noatime`

FS	Rand. write *(base → tuned)*	Video write *(base → tuned)*
Ext4	144.5 → 168.2	1196.9 → 1670.5
XFS	125.0 → 133.0	1239.3 → 1524.6
Btrfs	101.7 → 98.7	1243.5 → 1363.8

Filesystem-Specific Tuning

Profile	Rand. *MB/s*	Video *MB/s*
Ext4 + `data=writeback`	147.2	1332.7
Ext4 + `barrier=0`	169.5	1255.4
XFS + `nobarrier`	–	–

Takeaway

- `noatime`: observed gains appear in some write workloads, but these results do not establish strict causality.
- **Safety trade-off:** unsafe options such as `barrier=0` trade crash consistency for only modest gains.
- **CPU note:** gains are not explained by a uniform CPU drop; in tuned video write, **Btrfs** still carries much higher system CPU (12.46%) than Ext4/XFS (∼2.3–2.4%).

Final Conclusions

Main Findings

- **Ext4:** the best choice in this study for direct-I/O random write.
- **XFS:** the strongest sequential-write profile, and very competitive for concurrent streaming.
- **Btrfs:** best justified by features and semantics, rather than raw write efficiency.
- **Mail Server:** Ext4 and XFS are the stable and interpretable choices.

Overall Conclusion

There is **no single best filesystem**: workload pattern, semantics, and safety determine the right choice.