# Enterprise Filesystems

Eric Sandeen
~~Principal Software Engineer~~
Associate Manager
Red Hat
Nov 15, 2017

# What We'll Cover

- Local “Enterprise-ready” Linux filesystems
  - ~~Ext3~~
  - Ext4
  - XFS
  - ~~BTRFS~~
  - Overlayfs
- Use cases, features, pros & cons of each
- Recent & future work
  - Features past present & future
- Benchmarks



# Local Filesystems in RHEL7

- We ship what customers need and can rely on
- We ship what we test and support
  - Major on-disk local filesystems
    - Ext4, XFS
  - Special-case “filesystems”
    - OverlayFS
  - Others are available for special purposes
    - Ext3, fat, vfat, msdos, udf, cramfs, squashfs...
  - We'll cover the “big two” today



# Whither ext3?

- **Ext3 is available and supported, but not recommended (provided in ext4.ko)**
- Ext3 ~~is~~ was the most common file system in Linux
  - Most distributions historically used it as their default
  - Applications tuned to its specific behaviors (fsync...)
  - Familiar to most system administrators
- Ext3 challenges
  - File system repair (fsck) time can be extremely long
  - Limited scalability - maximum file system size of 16TB
  - Can be significantly slower than other local file systems
  - direct/indirect, bitmaps, no delalloc …



# The Ext4 filesystem

- Ext4 has many compelling features
  - Extent based allocation
  - Faster fsck time (up to 10x over ext3)
  - Delayed allocation, preallocation
  - Higher bandwidth
  - Should be relatively familiar for existing ext3 users
- Ext4 challenges
  - Large device support not as polished in user space tools
  - Red Hat tests & supports up to 50TB in RHEL7



# The XFS filesystem

- XFS is very robust and scalable
  - Very good performance for large storage configurations and large servers
  - *Many* years of use on large (> 100TB) storage
  - Red Hat tests & supports up to 500TB in RHEL7
- XFS challenges
  - Old myths (sometimes based in fact)
  - Historically not as well known by customers and field support people
  - Historically had performance issues with meta-data intensive (create/unlink) workloads



# Whither BTRFS?

- BTRFS is the “newest” major local file system
  - Btrfs is 10 years old
- Shipped in RHEL6, RHEL7 as a tech preview item
- No further updates in RHEL:

Btrfs has been deprecated

The Btrfs file system has been in Technology Preview state since the initial release of Red Hat Enterprise Linux 6. Red Hat will not be moving Btrfs to a fully supported feature and it will be removed in a future major release of Red Hat Enterprise Linux.

The Btrfs file system did receive numerous updates from the upstream in Red Hat Enterprise Linux 7.4 and will remain available in the Red Hat Enterprise Linux 7 series. However, this is the last planned update to this feature.



# The OverlayFS virtual / union filesystem

- Overlay one (or many) filesystems on another
- Changes recorded only in “upper,” “lower” is untouched
- Allows page cache sharing for all unmodified files
- Extremely useful for container deployment
  - New inodes/pages only for modified files
  - Significantly lowers memory footprint
  - Not … quite 100% POSIX compliant - subset
    - 2 opens, 1 RDONLY & 1 RDWR – get 2 different files
    - Rename(2) may fail, application must detect
- In RHEL7.1, enhanced in RHEL7.4



# Generic but interesting local fs features

- Ext4, XFS, both have:
  - Delayed allocation
  - Per-file space preallocation
  - Hole punch
  - Trim / discard
  - Barrier (now flush/FUA) support
  - Defragmentation
  - Project Quota support



# How to Choose?  Use Cases

- Ext4
  - Fine for general use, familiar
  - Reasonable performance, scalability somewhat limited
- XFS
  - “If you have large or lots, use XFS” - Valerie Aurora
  - Anything over 16T is probably wise
  - Don't fear the metadata!
- Overlayfs
  - Docker / containers
  - Uses either xfs or ext4 on disk



# RHEL limits

| | RHEL6 | RHEL7 |
|---|---|---|
| Ext3 filesystem size | 16T | 16T |
| Ext4 filesystem size | 16T | 50T |
| XFS filesystem size | 300T | 500T |

- Max *supported* filesystem size
- RHEL fs limits are tested limits, not theoretical limits



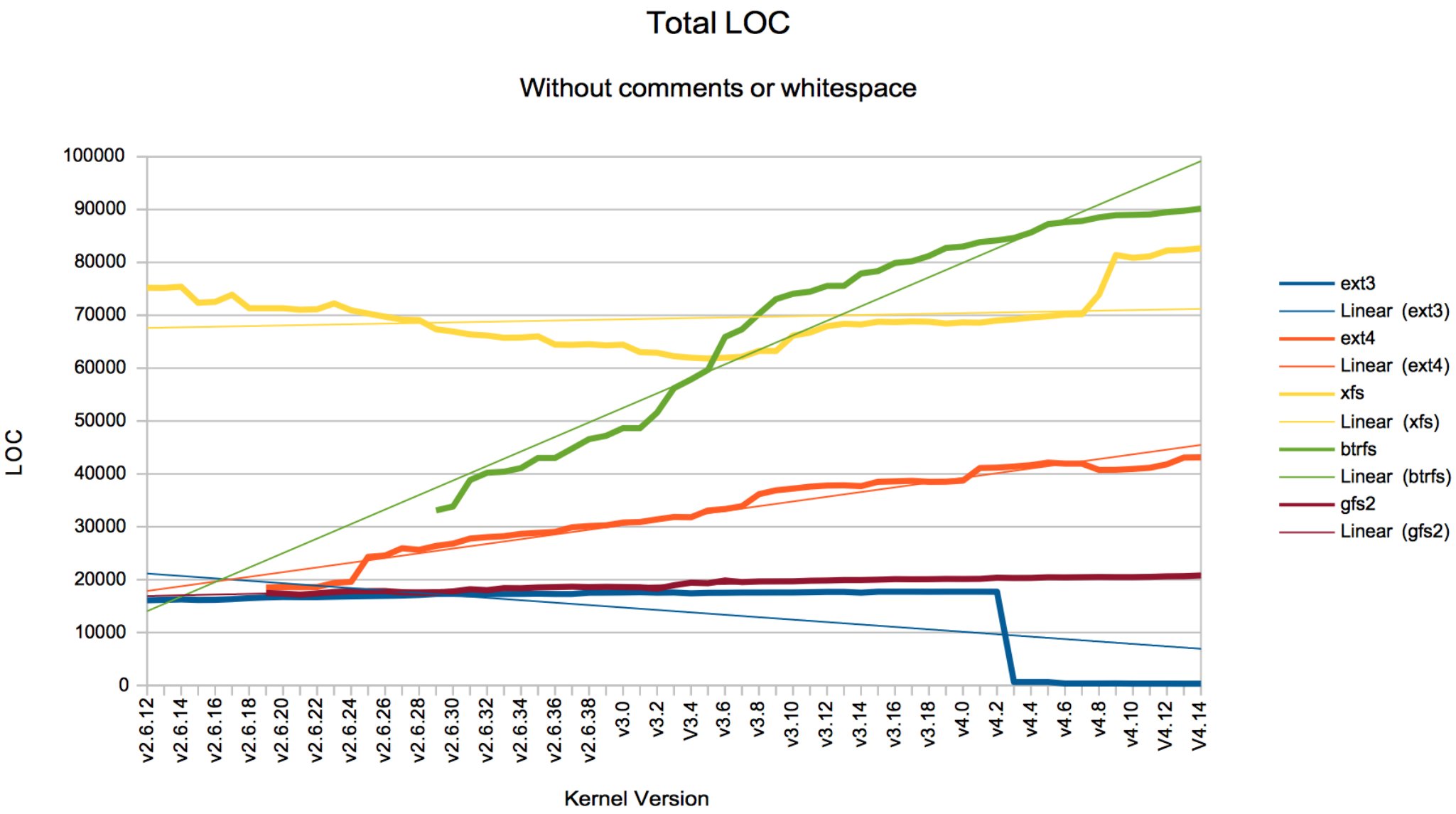
# Active Maintenance and Development

- From kernel v3.10 (~RHEL7) to v4.14 (today):
  - Ext4 : 1120 commits, ~116 authors
  - XFS : 1720 commits, ~164 authors
  - Btrfs : 2882 commits, ~202 authors
  - Overlayfs: 290 commits, ~33 authors
- Most are heavily weighted towards some authors, i.e. :

```
# git log --no-merges v3.10.. fs/xfs | grep ^Author | awk -F "<" '{print $1}' | sort | uniq -c |
sort -n | tail -n 8
     32 Author: Jan Kara
     34 Author: Al Viro
     58 Author: Jie Liu
    139 Author: Eric Sandeen
    197 Author: Brian Foster
    251 Author: Christoph Hellwig
    254 Author: Darrick J. Wong
    439 Author: Dave Chinner
```



# Active Maintenance and Development








# Where to now?

- All of these filesystems face challenges in the future
  - Ability to scale in sheer filesystem size
    - Disk format / containers mostly in shape
    - But structures & algorithms …?
  - Integrity with large storage
    - Detect errors from disk at runtime with checksums
    - On data?  On metadata?
    - Then what?
  - More Features!



# Ext3 Scaling & Features

- Nope.



# Ext4 Scaling & Features

- Bigalloc (since kernel v3.2)
  - Workaround for bitmap scalability issues
  - Allocates *multiples* of 4k blocks at a time
  - Not true large filesystem blocks, but close?
- Inline Data – (since kernel v3.8)
  - Store data inline in (larger) inodes
  - Mitigate bigalloc waste?
- Metadata Checksumming – (since kernel v3.5)
- Encryption (v4.0), Project Quota (v4.4)



# XFS Scaling & Features

- “Delayed logging” is done (v2.6.35)
  - dramatically improved metadata performance
  - default since v2.6.39
  - Last™ big performance issue
- File types in directories (finally!)
  - ls –color without stat
  - Overlayfs whiteouts
- Integrity work came next
  - CRCs on all metadata and log (v3.8)
  - FS UUID to detect misdirected writes (v3.8)



# XFS Scaling & Features

- Free inode btree (v3.16)
  - Fast free inode location and use
- Sparse Inodes (v4.2)
  - Efficient use of fragmented freespace
- Reflink (v4.9, “EXPERIMENTAL”)

```
< XFS has gained super CoW powers! >
 ---------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

  - Shared extents, copy on write, cp –reflink, dedupe, etc!
  - OverlayFS copy up, nfsv4 clone file range
  - On the cusp of losing “experimental” tag upstream



# XFS Scaling & Features

- Online scrub/repair (v4.15, “EXPERIMENTAL”)
  - May lead to online repair (!)
- New in-memory extent format lowers memory footprint
  - Avoids dreaded “memory allocation deadlock”



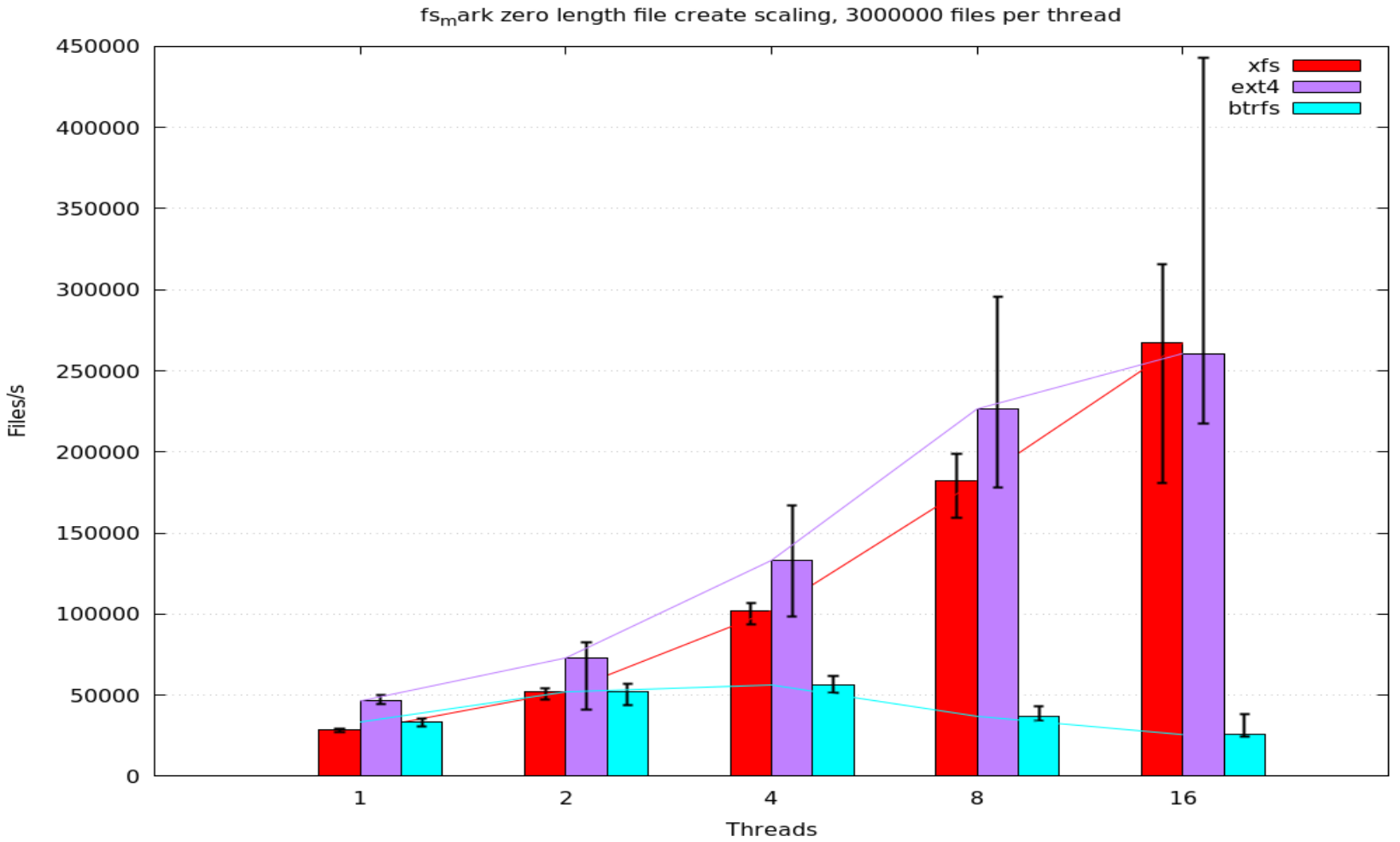
# Storage-related projects on the horizon

- Stratis project
  - Storage management framework & utility
- Springfield project
  - Storage management API
- NVDIMMs / DAX
  - Non-volatile memory storage
  - DAX – Page cache bypass for filesystems on NVDIMMs



# Lies, Damned Lies, and Benchmarks

- fs_mark zero length file create scaling, 3M/thread

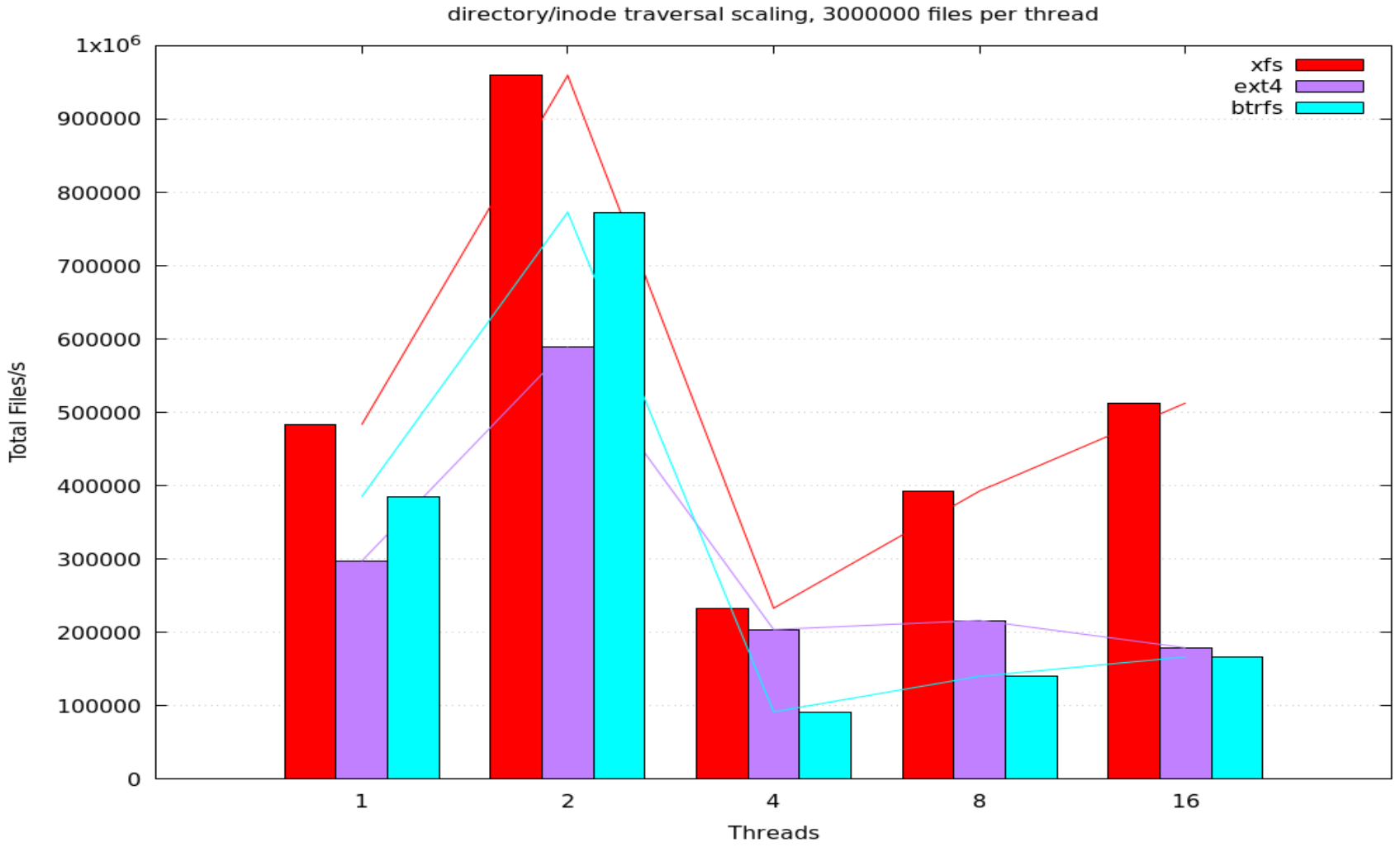







# Lies, Damned Lies, and Benchmarks

- fs_mark find scaling, 3M/thread

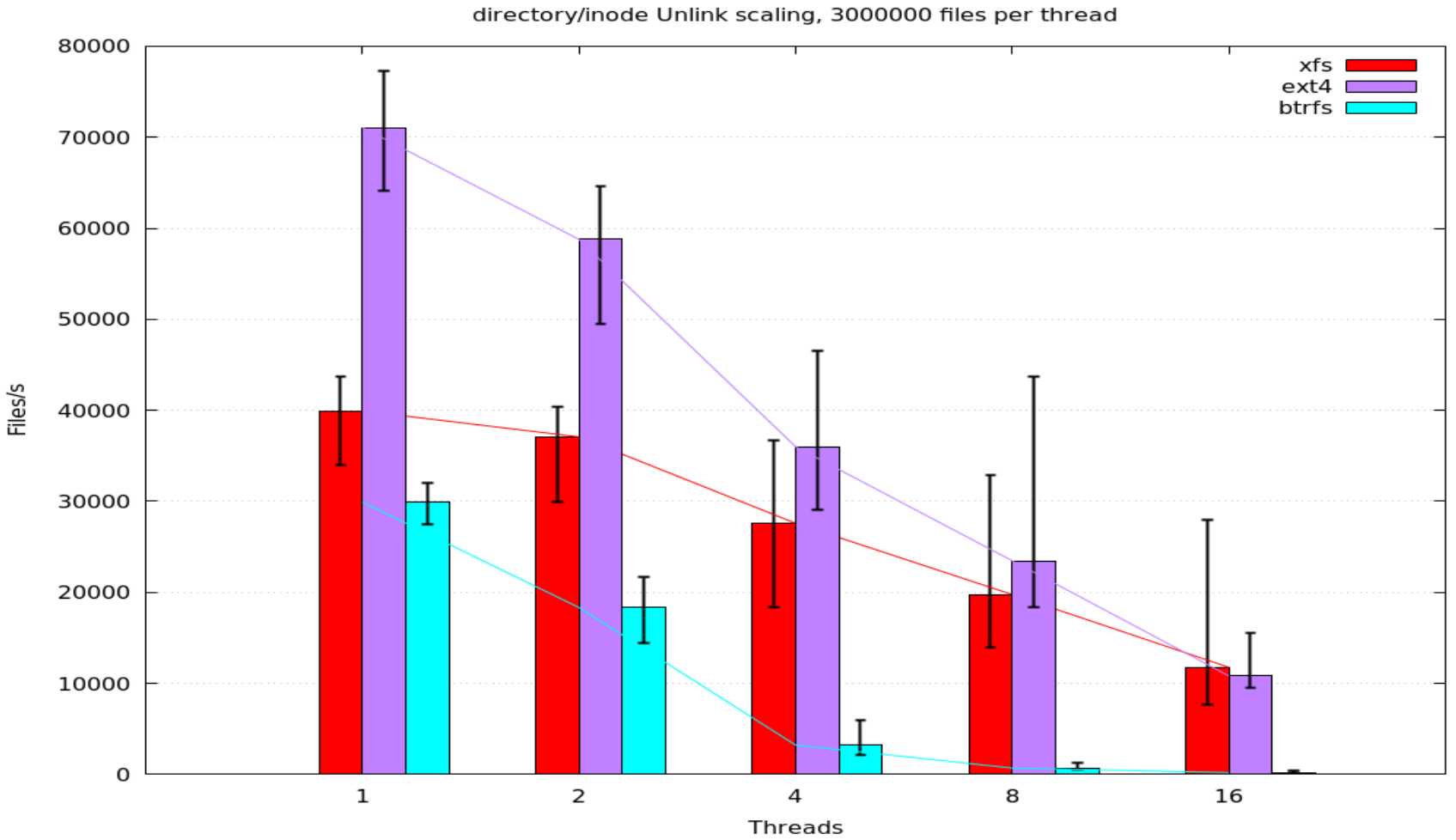







# Lies, Damned Lies, and Benchmarks

- fs_mark zero length file unlink scaling, 3M/thread

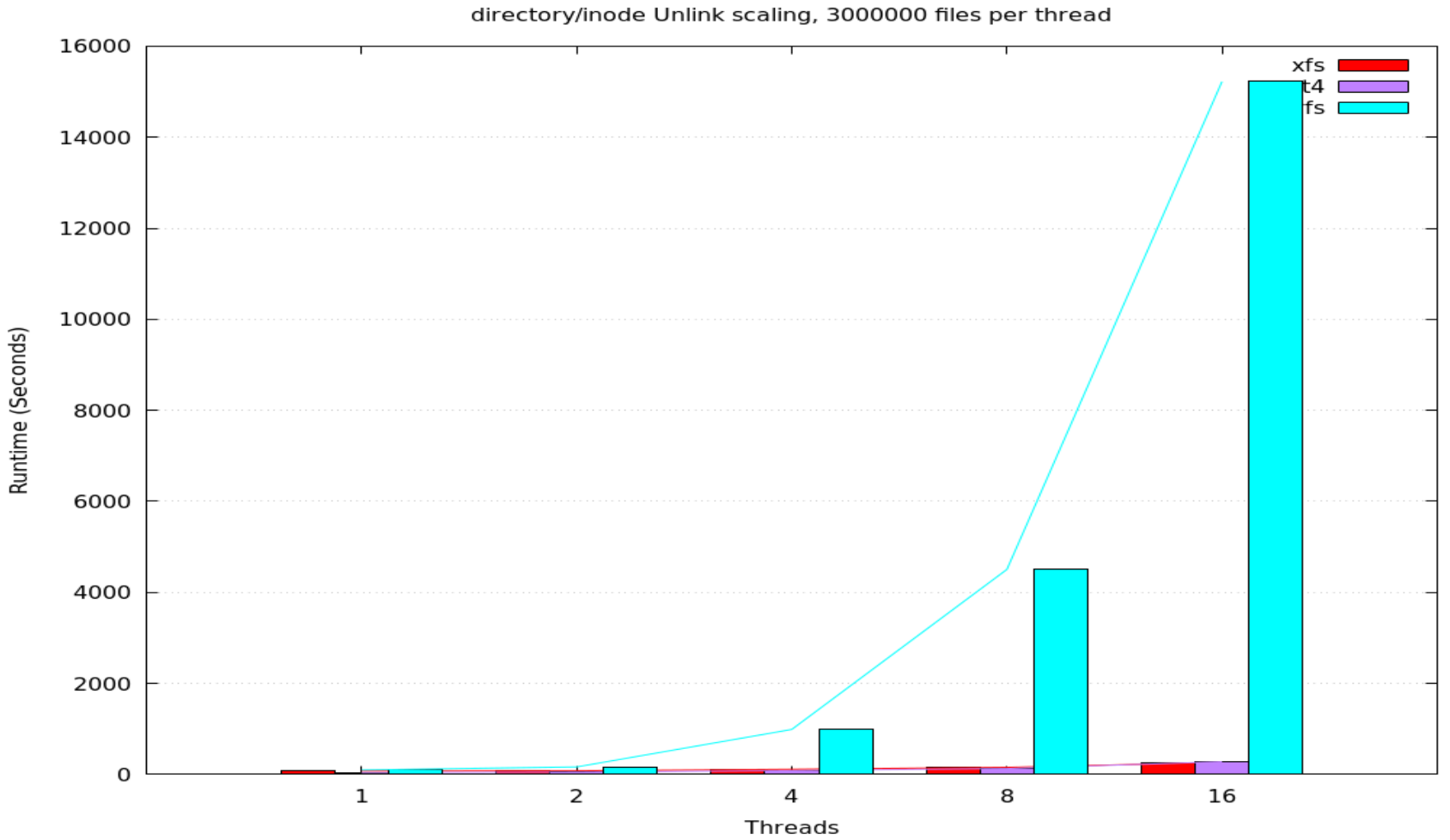







# Lies, Damned Lies, and Benchmarks

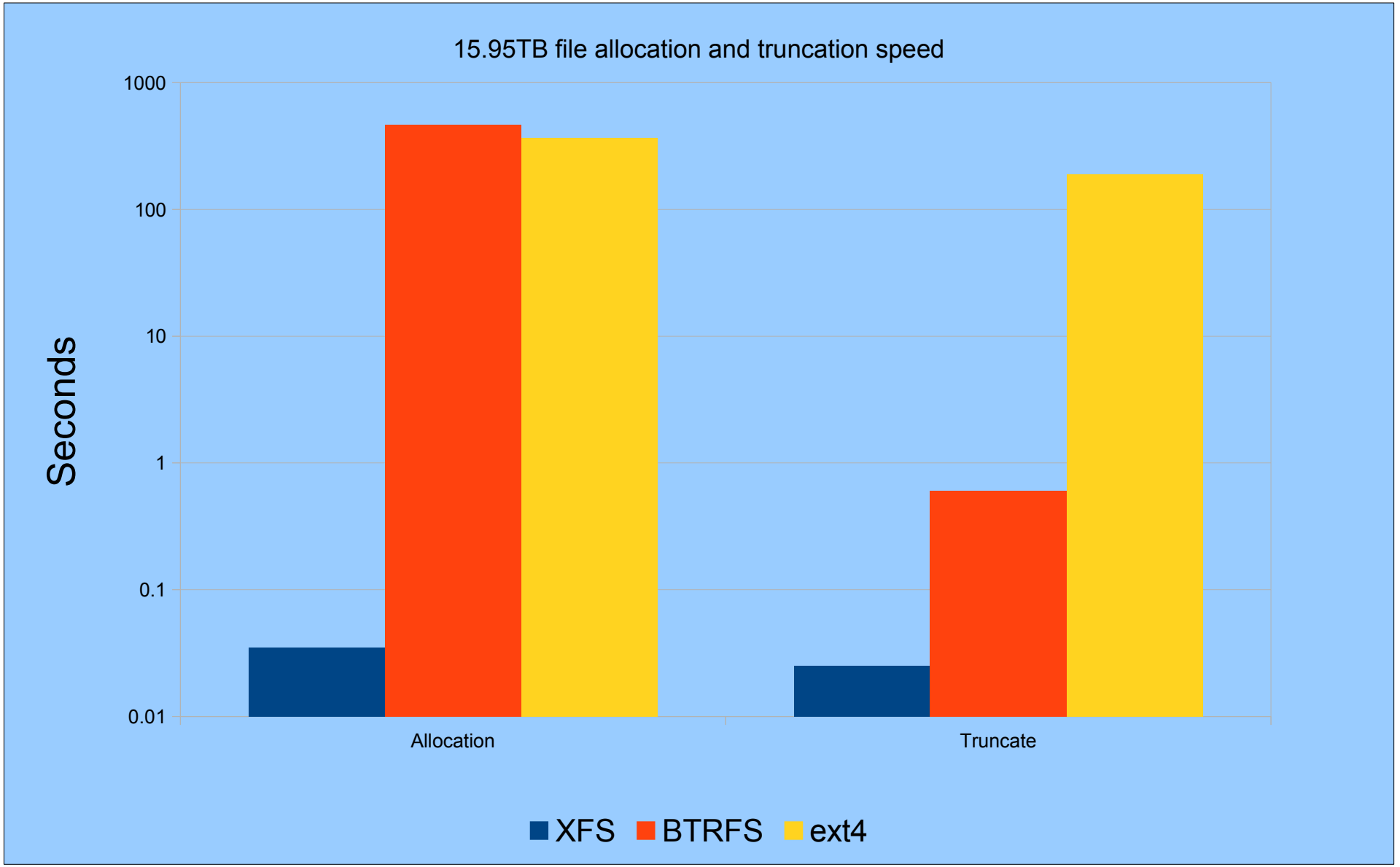
- fs_mark zero length file unlink scaling, 3M/thread








# Lies, Damned Lies, and Benchmarks






# Lies, Damned Lies, and Benchmarks

- enterprisestorageforum.com fsck test
- md RAID-60 on DDN LUNS; fs_mark population

| FS Size, TB | Nr of Files (millions) | XFS (seconds) | Ext4 (seconds) |
|---|---|---|---|
| 72 | 105 | 1629 | 3193 |
| 72 | 51 | 534 | 1811 |
| 72 | 10 | 161 | 972 |
| 38 | 105 | 710 | 3372 |
| 38 | 51 | 266 | 1358 |
| 38 | 10 | 131 | 470 |



# RHEL7 File System Updates

- RHEL 7.1
  - OverlayFS added
- RHEL 7.3
  - XFS CRCs on by default
  - Per-filesystem XFS statistics
- RHEL 7.4
  - SELinux on Overlayfs
  - Last BTRFS update
- XFS updated regularly in each



# RHEL Ongoing Work

- Ease of Use across storage administration
- Better interoperation with thinly provisioned storage
  - dm-thinp in particular
- NVDIMM & DAX support
- NVDIMM-native fs?  (NOVA?)



# Resources & Questions

- Mailing lists
  - xfs@oss.sgi.com
  - linux-ext4@vger.kernel.org
- IRC
  - #xfs on irc freenode.net
  - #ext4 on irc.oftc.net
- Web
  - https://stratis-storage.github.io/
- Questions?