

How LDISKFS Makes EXT4 Better (and Vice Versa)

Artem Blagodarenko, Ph.D.

Andreas Dilger



LDISKFS – An Enhanced EXT4 for Lustre

1. Copy ext4 from kernel

UPSTREAM

Linux Kernel

```
linux/  
├── fs/  
│   └── ext4/  
│       ├── super.c  
│       ├── inode.c  
│       ├── extents.c  
│       ├── mballocc.c  
│       ├── dir.c  
│       ├── xattr.c  
│       └── namei.c  
└── ...
```

2. Apply ldiskfs patches

TRANSFORM

Apply Patches

```
--- a/fs/ext4/super.c  
+++ b/fs/ldiskfs/super.c  
@@ -128,7 +128,19 @@  
     ext4_init_pageio();  
-    ext4_mb_init_backend();  
+    ldiskfs_mb_init_backend();  
+    ldiskfs_init_iam();  
+    #ifdef CONFIG_LDISKFS  
+    ldiskfs_register_super(sb);  
+    #endif  
     ext4_init_system_zone();
```

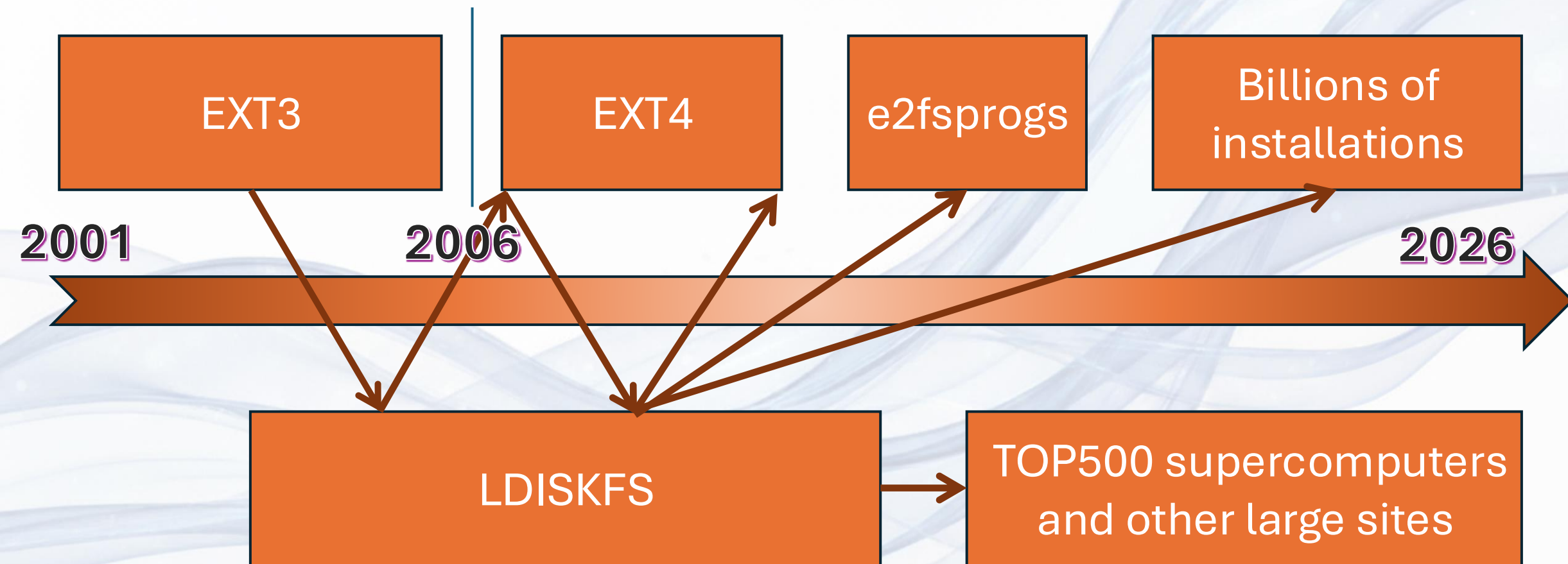
3. Build ldiskfs module

OUTPUT

LDISKFS Module

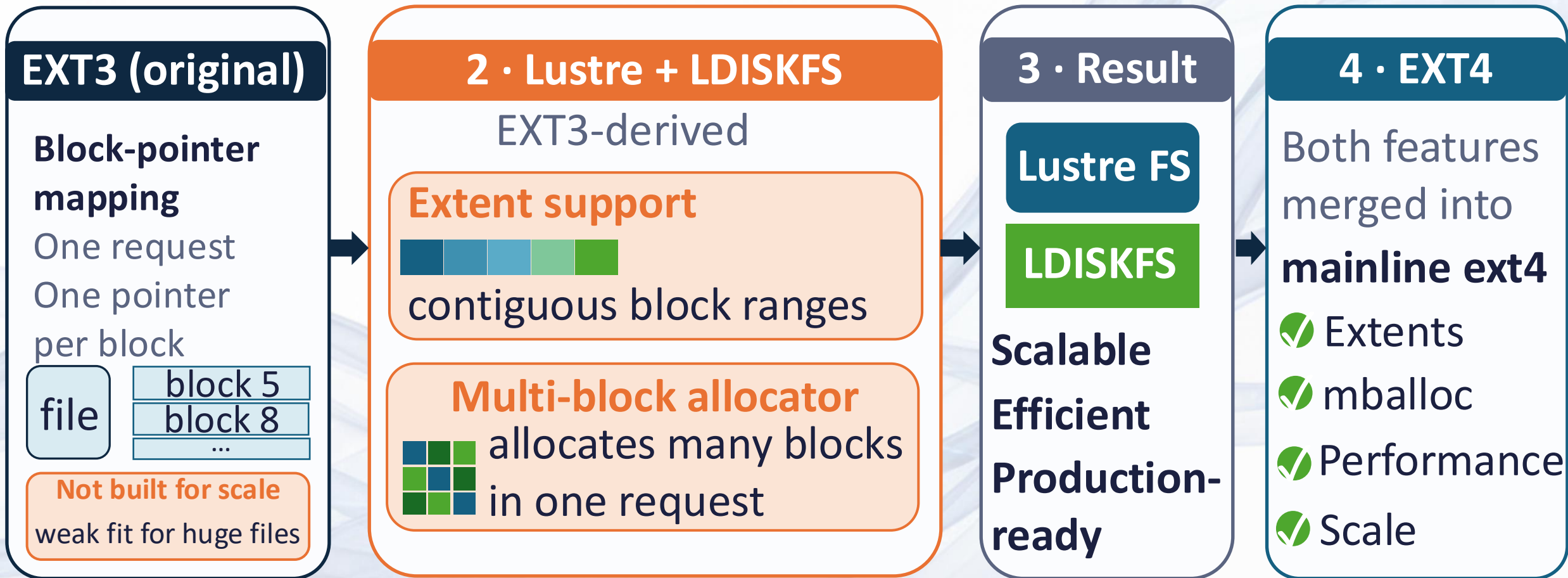
```
ldiskfs.ko  
+ parallel dir ops (pdiops)  
+ dirdata (FID in dirent)  
+ trusted.projid xattr  
+ nocmtime (MDT-managed)  
+ journal commit callback  
+ tunable preallocation
```

EXT4 Features, Utilities, Testing benefit from Ldiskfs



PAST

Back to EXT3 days



Lustre added extent support and multi-block allocation to ext3-derived LDISKFS — later upstreamed into ext4.

Extent File Layouts for Ext3

Alex (Tomas) Zhuravlev & Andreas Dilger designed/implemented extents

<https://www.kernel.org/doc/ols/2005/ols2005v1-pages-77-104.pdf>

Presented at 2005 Linux Symposium

The **metadata** describing the file becomes **dramatically smaller** when allocations are **contiguous**.

- **[RFC] file extents for EXT3**

From: Alex Tomas alex@clusterfs.com

Date: Aug 11, 2003

- **[RFC] extents, delayed allocation, mballoc for ext3**

From: Alex Tomas alex@clusterfs.com

Date: Apr 13, 2004

- **[PATCH 1/9] extents for ext4**

From: Mingming Cao cmm@us.ibm.com

Date: Aug 09, 2006

Signed-Off-By: Alex Tomas alex@clusterfs.com

Multi-Block Allocator (mballoc)

Allocate **multiple** filesystem **blocks at once** rather than making individual block allocation decisions and searches

Extent maps and delayed allocation – very important features that were developed at the same time, but only merged into ext4 in 2008

[PATCH] ext4: Add multi block allocator for ext4

From: Alex Tomas alex@clusterfs.com Date: Jan 29, 2008

Signed-off-by: Alex Tomas alex@clusterfs.com

Signed-off-by: Andreas Dilger adilger@clusterfs.com

Signed-off-by: Aneesh Kumar aneesh.kumar@linux.vnet.ibm.com

Signed-off-by: Eric Sandeen sandeen@redhat.com

Signed-off-by: "Theodore Ts'o" tytso@mit.edu

Multi-Block Allocator. Again

[RFC PATCH] don't search large block range if disk is full

From: Artem Blagodarenko

artem.blagodarenko@...il.com

Date: Mar 11, 2019

based on FS
condition

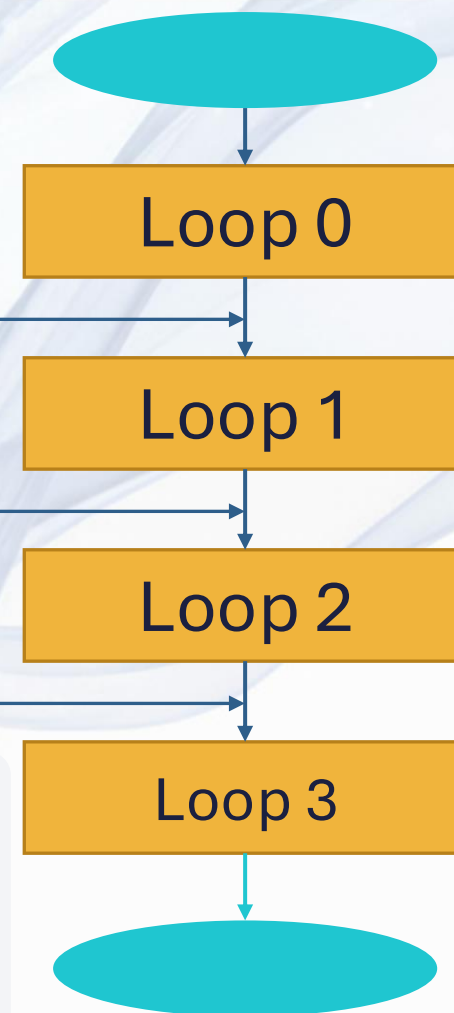
force to skip
useless loops

```
echo "75" > /sys/fs/ldiskfs/loop1/mb_c1_threshold
echo "85" > /sys/fs/ldiskfs/loop1/mb_c2_threshold
echo "95" > /sys/fs/ldiskfs/loop1/mb_c3_threshold
```

Start here if 75%
of disk is filled

Start here if 85%
of disk is filled

Start here if 95%
of disk is filled



Lustre developers provide features and do life-time support for them

Multi-Block Allocator Optimized. Finally

Solves problem of "useless allocator loops". Find needed chunk size for $O(1)$ on CR0 and $O(\log n)$ on CR1

Subject: Block allocator improvements Date: Mar 15, 2021

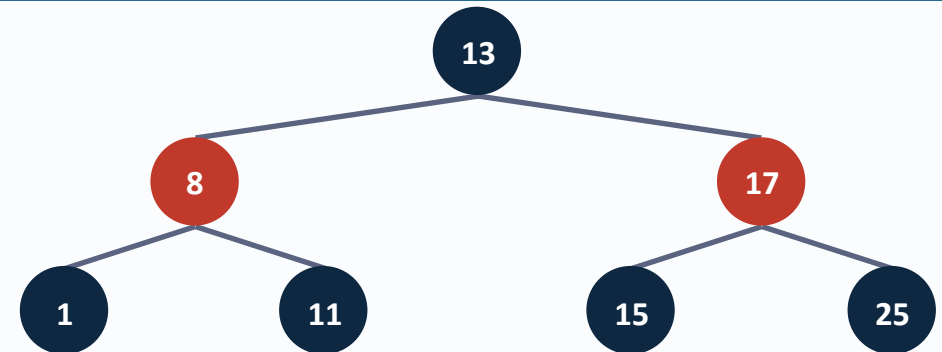
From: Harshad Shirwadkar harshadshirwadkar@gmail.com

CR0 · array of lists by order

2^0 : Group N — Group M ...
 2^1 : Group K — Group L ...
 $\vdots$
 2^{32} : Group N — Group M ...

One list per order $\rightarrow O(1)$ lookup

CR1 · red-black tree by fragment size



Sorted by largest fragment $\rightarrow O(\log n)$



LDISKFS still grows

As drive size dramatically increases

...8TB HDD → 60TB → **245TB NVMe**



Maximum backend storage size increases

...16TB → **2PB+**



LDISKFS quickly exceeded
the original design!

Extending max partition size

- LDISKFS doesn't allow to create target with size larger than verified
- To ignore this limitation there an “force_over_256tb” option
- Every time Lustre FS needs to have a new verified limit, scale testing is done and this limit is moved forward

force_over_1024tb	LU-12950	5 March, 2020	2.14
force_over_512tb	LU-8974	10 Feb, 2017	2.10
force_over_256tb	LU-7592	22 April, 2016	2.9
force_over_128tb	LU-136	08 July, 2011	1.8
force_over_16tb	b=16038 (Bugzilla)	16 July, 2010	1.6
force_over_8tb	b=17659 (Bugzilla)	29 June, 2009	1.6

[Scaling LDISKFS for the Feature](#) and [Scaling LDISKFS for the future. Again](#)

Ext4 MBalloc on Hybrid IOPS Device (LU-16750)

Combine NVMe and HDD, or TLC and QLC into **one** ldiskfs filesystem

- Block DM device created with fast storage at start, slow storage at end
- Hybrid filesystem allocates ldiskfs metadata in fast "IOPS" groups
- Mballocc efficiently switches allocations between group types
- Statically allocated superblocks, bitmaps, inode table, journal
- Dynamically allocated directories, xattrs also use reserved groups

Subject: ext4: optimize metadata allocation for hybrid LUNs

From: Zhenyu (Bobijam) Xu bobijam@whamcloud.com

Date: Sept 12, 2023



Delayed Allocation

Postpones choosing physical disk blocks,
merge writes before data block allocation

Larger block ranges, fewer calls

Subject: [RFC] delayed allocation for ext4

Alex Tomas (Zhuravlev) bzzz@clusterfs.com

Date: Dec 22, 2006

Subject: 2.6.25-rc1-ext4 patchset released

From: Theodore Ts'o Date: Feb 10, 2008

Alex Tomas (Zhuravlev) bzzz@clusterfs.com

vfs: add basic delayed allocation support
ext4: Add basic delayed allocation support

LDISKFS support is still in progress ([LU-12916](#))



FIEMAP - File Extent MAP

Efficient interface to dump block allocation layout of a file – filefrag
Answers "*Is my file fragmented on disk?*" even for Lustre striped files
Check if blocks contiguously allocated, delayed, encrypted, offline
Lustre FIEMAP design June 2007

by Andreas Dilger and Kalpak Shah

Subject: **Fiemap, an extent mapping ioctl**

From: Mark Fasheh mfasheh@suse.com

Date: Jun 25, 2008

Subject: **vfs: vfs-level fiemap interface**

Date: October 8, 2008

Signed-off-by: Mark Fasheh mfasheh@suse.com

Signed-off-by: Theodore Ts'o tytso@mit.edu

Includes significant input from

Kalpak Shah kalpak.shah@sun.com

Andreas Dilger adilger@sun.com

Eric Sandeen sandeen@redhat.com

Mark Fasheh mfasheh@suse.com



Large xattrs / ea_inode

Efficient storage for xattrs inside large inode

Heavily used by MDT for Lustre layout, FID, ...

Subject: **Large EAs for ext4**

Date: Nov 18, 2008

From: Kalpak Shah Kalpak.Shah@sun.com

[PATCH 01/32] ext4: xattr-in-inode support

Date: Wed Jun 21 2017 - 17:30:19 EST

From: Andreas Dilger andreas.dilger@xxxxxxxxxx

Changes suggested by Mingming and Andreas for earlier patches have been incorporated in this version.

2003–2005
ClusterFS
Alex Tomas + Andreas Dilger

EA-in-inode

EXT3 large inodes

2008

Lustre / ldiskfs

Large EA / external EA inode

2017

Upstream EXT4

EXT4_FEATURE_INCOMPAT_EA_INODE

2017/2018

e2fsprogs

ea_inode support

Project quota

Project quota is a way to account for and limit filesystem usage by a **project ID (projid)** rather than by user or group. Implemented for EXT4 by Lustre developers for EXT4 and then backported

- **2013 · LU-4017** — Lustre needs directory quotas: XFS has them, EXT4 doesn't
- **2014–2015 · ext4 first** — Li Xi (DDN) posts "add project quota support" v1 → v15 in public review. `i_projid` in the inode, `PROJINHERIT`, `PRJQUOTA` in the VFS, XFS-compatible ioctls.
- **2016 · Linux 4.5 + e2fsprogs 1.43** — merged as generic VFS + ext4 code Lustre-motivated, ext4-native.
- **2015–2017 · back to LDISKFS** — upstream code backported to RHEL7
- **Today** — one small patch left (`trusted.projid xattr` for MDT backup)
This extension itself does not change the on-disk format

largedir: Millions of Names in a Single Directory

Allow workloads that use very wide and flat namespaces

- **2011 ([LU-50](#)):** Lustre needed directories above ext4's ~10M entry limit. Liang Zhen's pdirops patch bundled in Lustre's own N-level htree extension, EXT4_FEATURE_INCOMPAT_LARGEDIR, raises limit to 2B+
- **2015 ([LU-6030](#) / [LU-6824](#)):** Split into its own ext4-large-dir.patch. Warning patch added while waiting on upstream/e2fsprogs support.
- **2017-09 In Linux ext4 4.13:** largedir feature (3-level htree, 2B+ entries, 2GB+ directories) merged into ext4 — authored and merged by Artem Blagodarenko, the headline feature of that merge window.
- **e2fsprogs 1.44.0 (Mar 2018):** `mke2fs -O large_dir` support ships
- **2019 ([LU-11922](#) / [LU-11546](#)):** LDISKFS reconciled with upstream feature

With DNE striped directories, can handle 100B+ entries in one directory

Parallel e2fsck: Scanning Petabyte Partitions

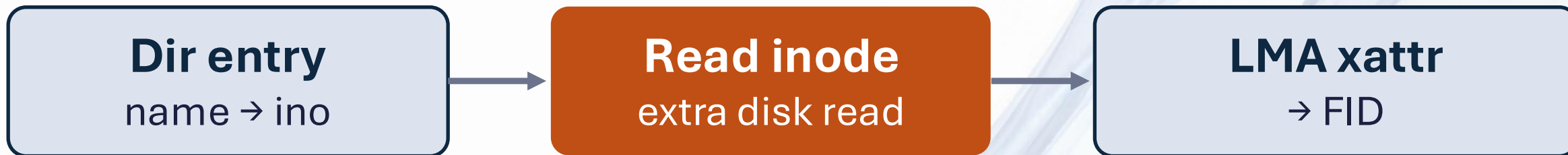
Handle ext4 filesystems with billions of files and PB of capacity

- **2016 ([LU-8465](#)):** e2fsck doesn't scale to PB-class Lustre targets
Pass1 alone: 3,771s on a 1PB/105M-inode OST (~95% of check time)
Proposed async I/O and parallelization of checks
- **2016–2020:** Wang Shilong writes, submits parallel e2fsck Pass 1/5 RFC, 50–70+ patches to linux-ext4
- **Result: ~40× speedup on Pass1, ~36× on Pass5** for large OSTs
- **2020 ([LU-8465](#) fixed in Lustre 2.14.0):** fixed in Whamcloud's e2fsprogs resolved a real MMP race under load ([LU-14126](#))
- **2026 ([LU-14894](#)):** Dongyang Li dongyangli@ddn.com writes parallel Pass 2
Pass2 directory scan currently longest phase for MDT filesystems
- **Still open ([LU-14213](#)):** use parallel e2fsck by default, not enabled today

PRESENT

Dirdata: Name → FID Lookup

Without dirdata



With dirdata



readdir: one inode block read (4KiB) saved per directory entry

Dirdata to EXT4

- **Dirdata: 17 years from Lustre hook to EXT4 feature**
- **2009 · born as "FID in dirent"** — Pravin Shelar & Andreas Dilger, Lustre 2.0. Store Lustre File IDentifier after filename, flag in file_type bits.
- **2017–2018 · first upstream attempt** — proposed as carrier for 64-bit inode numbers (v4 kernel + e2fsprogs series). Not merged: "no interest" beyond Lustre.
- **2026 · rewritten as a general facility** — three patches, self-describing records, unknown types skipped. First Lustre FID; second fscrypt + casefold hash. Sent to ext4 with an xfstest; e2fsprogs separate.

64-Bit Inode Numbers: Breaking the 4-Billion-Inode Ceiling

limit

$2^{32} \rightarrow 2^{64}$

~4 billion inodes / MDT
MDTs run out of inodes

dirent

low 32 bits inum	name\0	high 32 bits inum
------------------	--------	-------------------

└────────── same as today ─────────┐ **dirdata record**

result

Transparent upgrade: no reformat

The feature needs DIRDATA first

64-bit Inode Numbers for EXT4

- **2017 • proposal on linux-ext4**

- March: Blagodarenko: "64-bit inode number" thread
Dilger: dirdata can carry the high 32 bits of inode number
Ts'o: make it a feature flag to maintain compatibility
- Nov: RFC v2 kernel (dirdata + 64-bit inode)
e2fsprogs v2 (7 patches, Dilger / Blagodarenko / Bobi Jam).

- **2018 • full series**

- Feb: kernel v4, EXT4_INCOMPAT_INODE64, s_inodes_count_hi, high 32-bit as dirdata entry
- Mar: e2fsprogs v3 and v4, May: ext2_ino64_t, EXT4_INCOMPAT_64INODE. 64-bit kernels only

- **Outcome** — not merged; blocked on dirdata itself being out of tree.

- **2026 • plan for the future**

- New dirdata series reserves INO64 flag as the second record type.
- Once dirdata lands, 64-bit inodes add a dirent record
- Do not need a massive format change, both inode sizes co-exist in single dir
- Expand filesystem beyond 32-bit inodes

FUTURE

Lustre FS Raises the Bar

Lustre FS developers joining EXT4 weekly call:

- Can ext4 mount a 2PB filesystem?
 - I think ldiskfs can already do this today.
- Can ext4 have 100 billion files in the same partition?
 - We have a plan for that, let's discuss.
- How to quickly check 1PB partition correctness?
 - Lustre e2fsprogs already has parallel e2fsck.
- How to store 1B files in a single directory?
 - 3-level htree is already merged, give it a try.

Who will ever need this scale?

- Lustre FS today, ext4 tomorrow.

What's Next for LDISKFS and EXT4?

- Reduce EXT4 vs LDISKFS delta, switch to upstream EXT4
- This may be an overly optimistic and ambitious goal
- If we could achieve it, it would have several benefits:
 - Lustre developers working on LDISKFS could focus on EXT4
 - EXT4 would be better suited for large-scale use, not only Lustre
 - It would be easier for Lustre to maintain and support its backend filesystem

Questions?

