SEAGATE

Scaling LDISKFS for the future

Artem Blagodarenko

LUG 2016
Portland, Oregon

Lustre FS Backend Storages

ZFS

- ✓ *Developed by Sun Microsystems*
- ✓ *Has large scalability*
- ✓ *Fully asynchronous support*
- ✓ *Many other features like COW*

Lustre uses two file systems for backend storage

LDISKFS

- ✓ *Created as fast backend storage*
- ✓ *based on ext4 which has a large community*
- ✓ *There are many successfully deployed systems based on LDISKFS*

LDISKFS Partitions in Production

As drive size increases

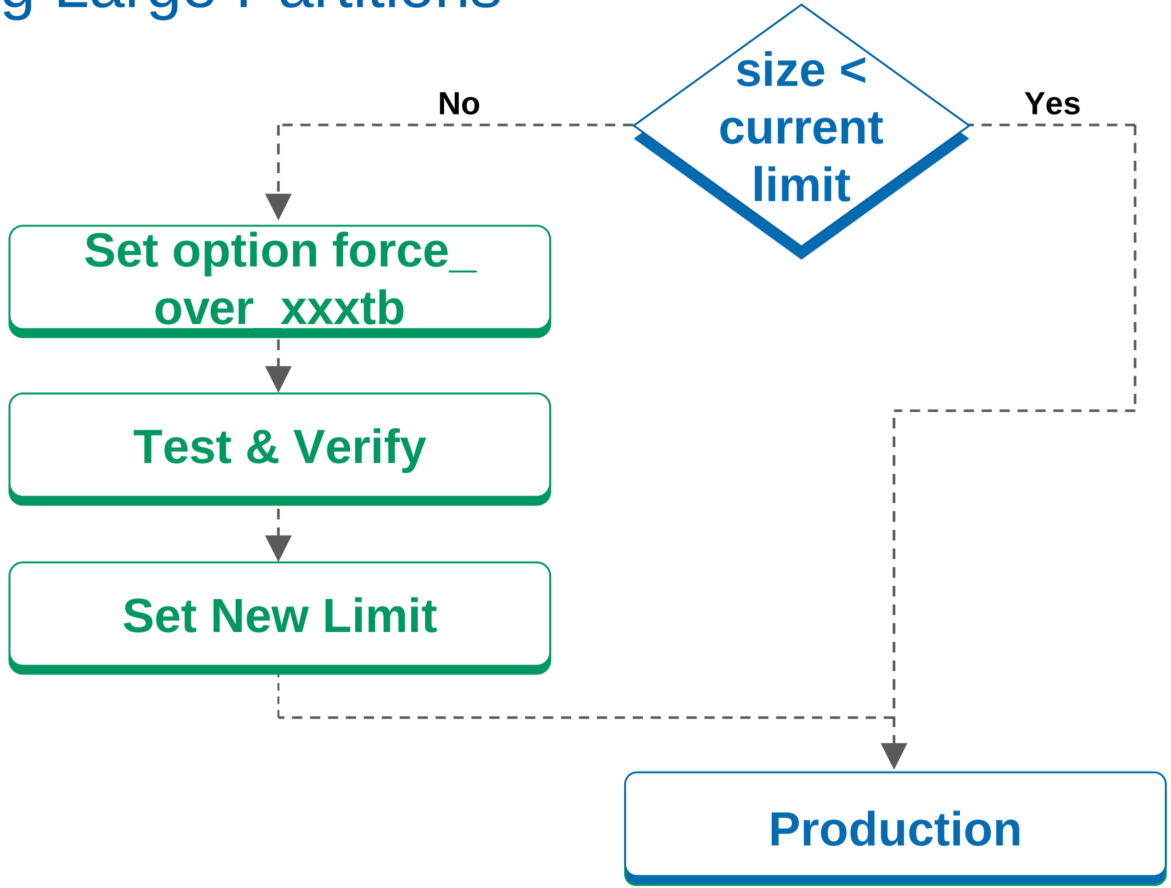
6TB > 8TB > 10TB

The maximum backend storage size increases

16TB > 300TB

LDISKFS quickly exceeded the original design!

Creating Large Partitions


size < current limit

No

Yes

Set option force_ over_xxxtb

Test & Verify

Set New Limit

Production

LDISKFS Max Size Increasing Challenges

- **Are formatting parameters still actual?**
- **Are Lustre FS structures ready?**
- **Are LDISKFS data structures ready?**
- **Has anyone tried to create such large partitions?**
- **Are Lustre FS Tools ready?**

MKFS Default Parameters

```
mkfs_cmd = mke2fs -j -b 4096 -L testfs-OSTffff  -J size=400 -I
256 -i 1048576 -q -O
extents,uninit_bg,dir_nlink,huge_file,64bit,flex_bg -G 256
-E lazy_journal_init,lazy_itable_init=0 -F /dev/md1
```

Inodes Count per Bytes Rate

EXT4 limited with UINT32_MAX (2^32-1=4294967296) inodes by design

Average file size *can be set using -i option or default value is used*

Partition Size	>10GB	>1TB	>4TB	>16TB
Average File Size	64kB	256kB	512kB	1MB

Set average file size for large partition	› 169 TB partition › -i 1048576 › 177635072 inodes *the smallest value is* **43368**
Use many OSTs	4294967296/177635072 = **24 OSTs**

Performance near first and last block of disk

- *Due to large disk size performance loss at the end of surface is possible*
- *There are mkfs options that move some metadata to optimal part of disk* **(flex_bg and -G)**
- *This options are currently used in our standard configuration, but numbers should be corrected*
- *This parameter could be adjusted for new size of disk. Option can be changed after* **LU-6442**

Scalability issues

- *Two level hash tree and leaf block which consist up 300 directory entry*
- *Total number of directory entry limited by ~20 millions entries*
- *Hash collisions decrease real directory size*
- **Solution**: *increase a hash tree levels, patch e2fsprogs and allow set special flag to FS (LU-7932)*

- **Use case:** *large number of* **creations + unlinks**.
- *Hash tree is created, but never reduced*
- *Large hash range assigned to one dir entry block and none free blocks in tree to split due long time usage*
- *Limited with 300 entry per directory if hash from name will bad*

LDISKFS data structures

Ext4 uses **ext4_fsblk_t** type for global block accessing and **ext4_lblk_t** for file logical blocks. ldiskfs patches use the same types.

```
typedef unsigned long long
ext4_fsblk_t;
```

```
/* data type for file logical block number */
typedef __u32 ext4_lblk_t;
```

```
/* data type for block offset of block group */
typedef int ext4_grpblk_t;
```

ext4_map_inode_page

There is function with parameter “unsigned long *blocks”:

```
int ext4_map_inode_page(struct inode *inode,
struct page *page,unsigned long *blocks, int create)
```

But **ext4_bmap** returns **sector_t** value.

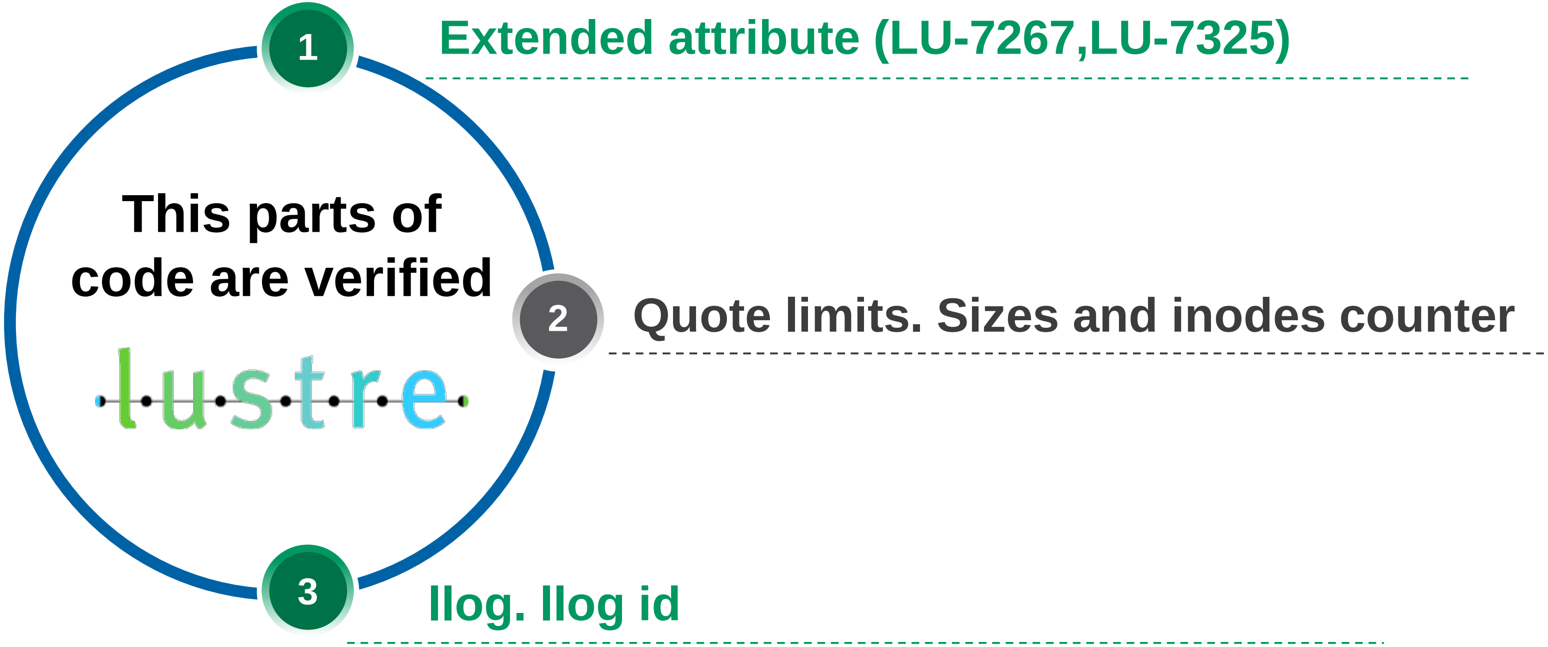
```
static sector_t ext4_bmap(struct address_space *mapping, sector_t block)
blocks[i] = ext4_bmap(inode->i_mapping, iblock);
```

That depending on macros can be 32 or 64 bit long

This issue is actual for x86_32 systems only, because unsigned long is 64bit long on x86_64 systems.

Patch landed in **LU-6464**!

Lustre FS structures



This parts of code are verified

lustre

1 Extended attribute (LU-7267,LU-7325)

2 Quote limits. Sizes and inodes counter

3 llog. llog id

Tools. FSCK

There are 64 bit for addressing blocks by number.

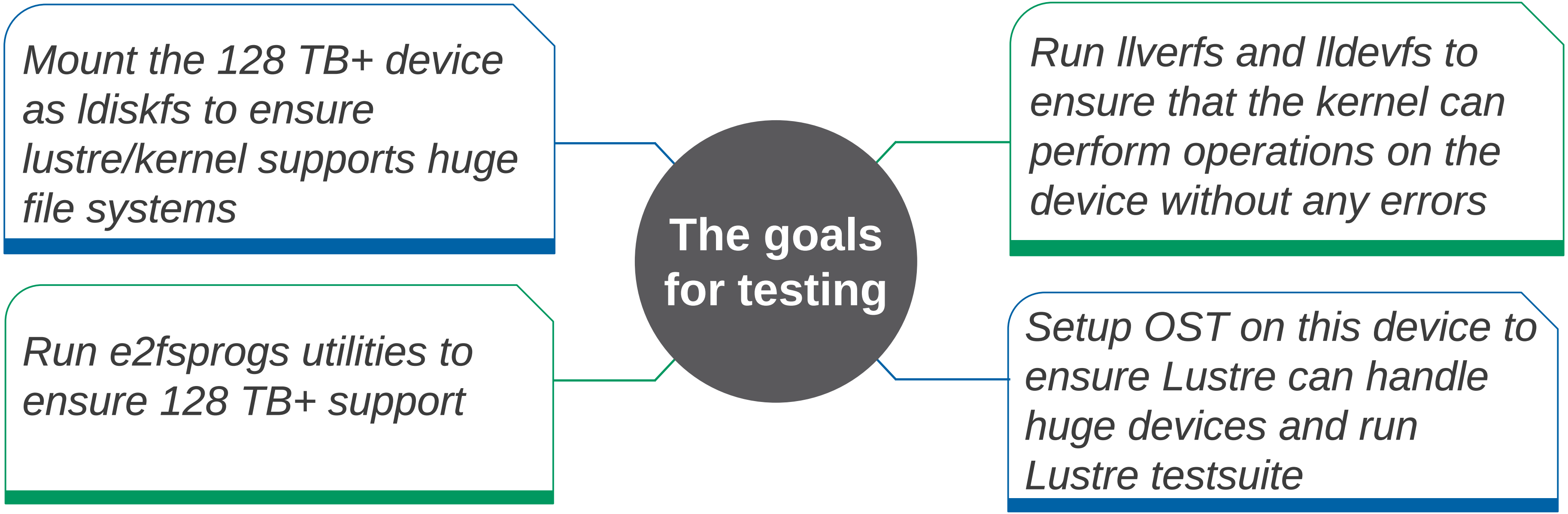
```
typedef __u64 __bitwise        blk64_t;
```

There is 32 bit version

```
typedef __u32 __bitwise        blk_t;
```

- *Bad blocks accessing in wrong way*
- *Some functions in bitmap layer uses blk_t*
- *Hurd translators*

Common Tests for 128 TB+ ldiskfs Partitions



The goals for testing

Mount the 128 TB+ device as ldiskfs to ensure lustre/kernel supports huge file systems

Run e2fsprogs utilities to ensure 128 TB+ support

Run llverfs and lldevfs to ensure that the kernel can perform operations on the device without any errors

Setup OST on this device to ensure Lustre can handle huge devices and run Lustre testsuite

Components To Be Tested		
e2fsprogs	*ldiskfs*	*Lustre*

Special test cases

MKFS lazy init

Testing inode allocation

New file in group at the end of the disk

Test for 64k subdirectory limit

Results

For Community

- *Code review*
- *Testing suite*
- *Patches with fixes*
- *Move LDISKFS size limit to* **256TB (LU-7592)**

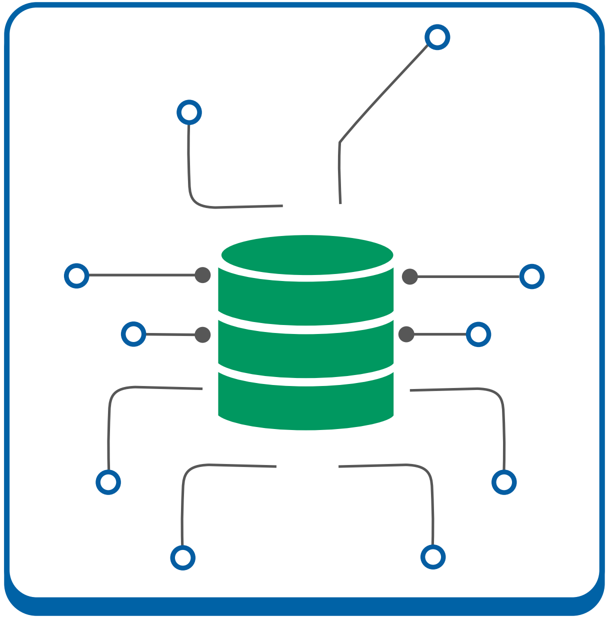
For Customers

- *Fewer OSTs*
- *Larger OSTs*
- *Decreased resource requirements*
- *Denser storage*

Current Work

Current work is focused on extending the limit above 256TB.

Future Work



Extending inodes count over UINT32_MAX

Check large memory blocks allocation

Invent solutions for large directories

Acknowledgments

Thanks to

Alexey Lyashkov
Elena Gryaznova
Meghan McClelland

SEAGATE

Thank you!