An Overview of the SquashFS filesystem

Phillip Lougher

SquashFS beginnings

- Working on Digital TV in 2001
  - Used Cramfs as an initrd filesystem
  - Wanted better compression
  - More inode attributes
    - Proper guid/uid
    - Proper timestamps
  - Think mountable “tar.gz”

SquashFS beginnings (cont)

- First release 23rd October 2002
  - Working name “cram2fs”, changed to SquashFS one week before release...
  - Mainly intended as an embedded rootfs. Hoped it might be used for archiving
  - Interestingly enough didn't think about liveCD usage
    - Knoppix probably only liveCD at the time

SquashFS beginnings (cont)

- Good points
  - Metadata compression
  - Maximum datablock size 32 KiB
  - Full uid/gid entries
  - Date stamps
    - Only on files and directories
    - Not on symlinks, device nodes

SquashFS beginnings (cont)

- “Bad” points
  - Limited to total of 48 uids & 16 gids
  - Maximum metadata size of 16 MiB each for inodes and directories
  - Maximum directory size 512 KiB
  - Maximum file size 4 GiB
  - Bit fields used to pack on disk structures

Bit field example

```
typedef struct {
    unsigned int        inode_type:4;
    unsigned int        mode:12;
    unsigned int        uid:4;
    unsigned int        guid:4;
    time_t              mtime;
    squashfs_block      start_block;
    unsigned int        file_size;
    unsigned short      block_list[0];
} __attribute__ ((packed))
squashfs_reg_inode_header;
```

SquashFS layout changes

Five layout changes in 24 releases over 6 years...

2.0 - 21st May 2004

2.1 - 10th December 2004

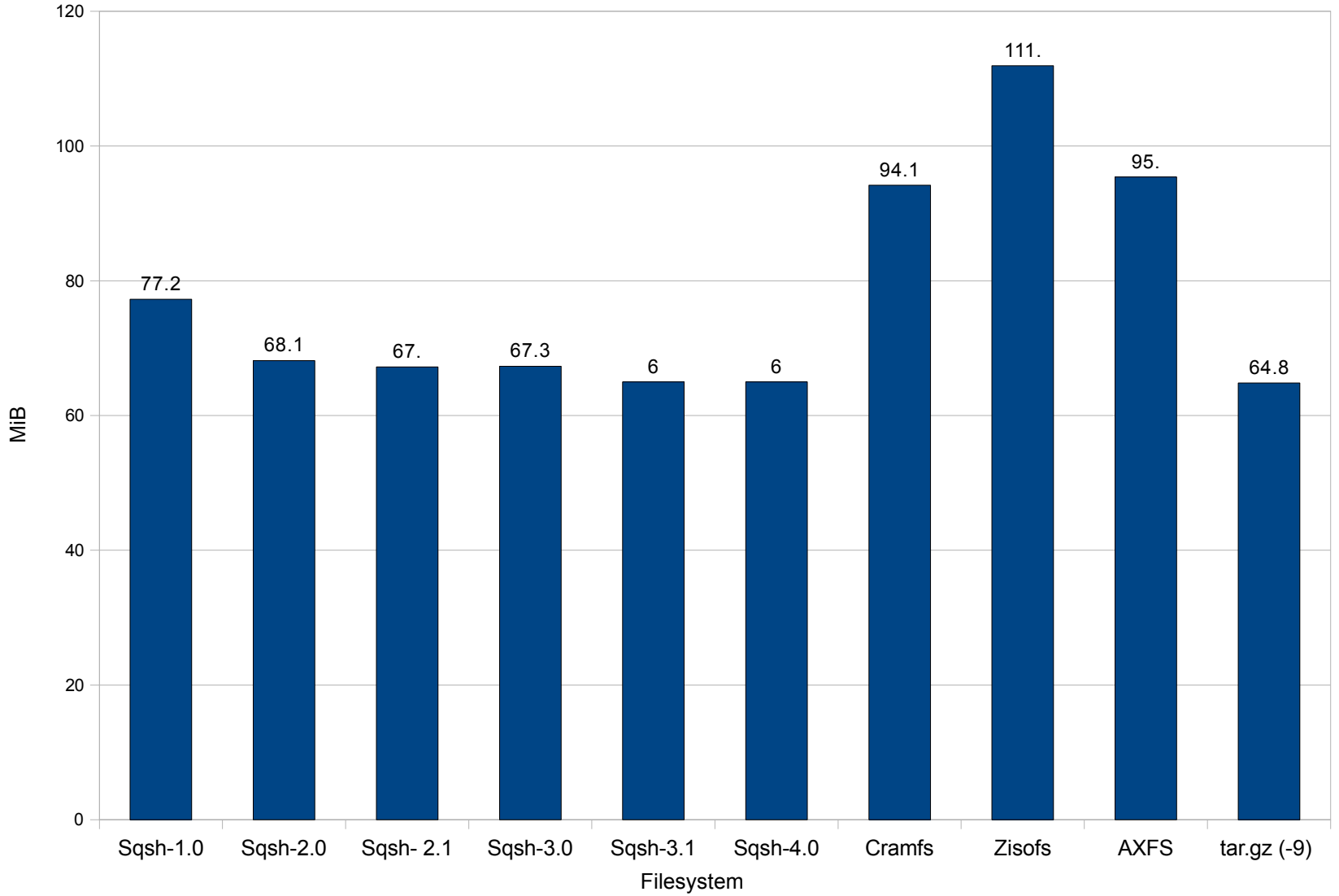
3.0 - 15th March 2006

3.1 - 1st November 2007

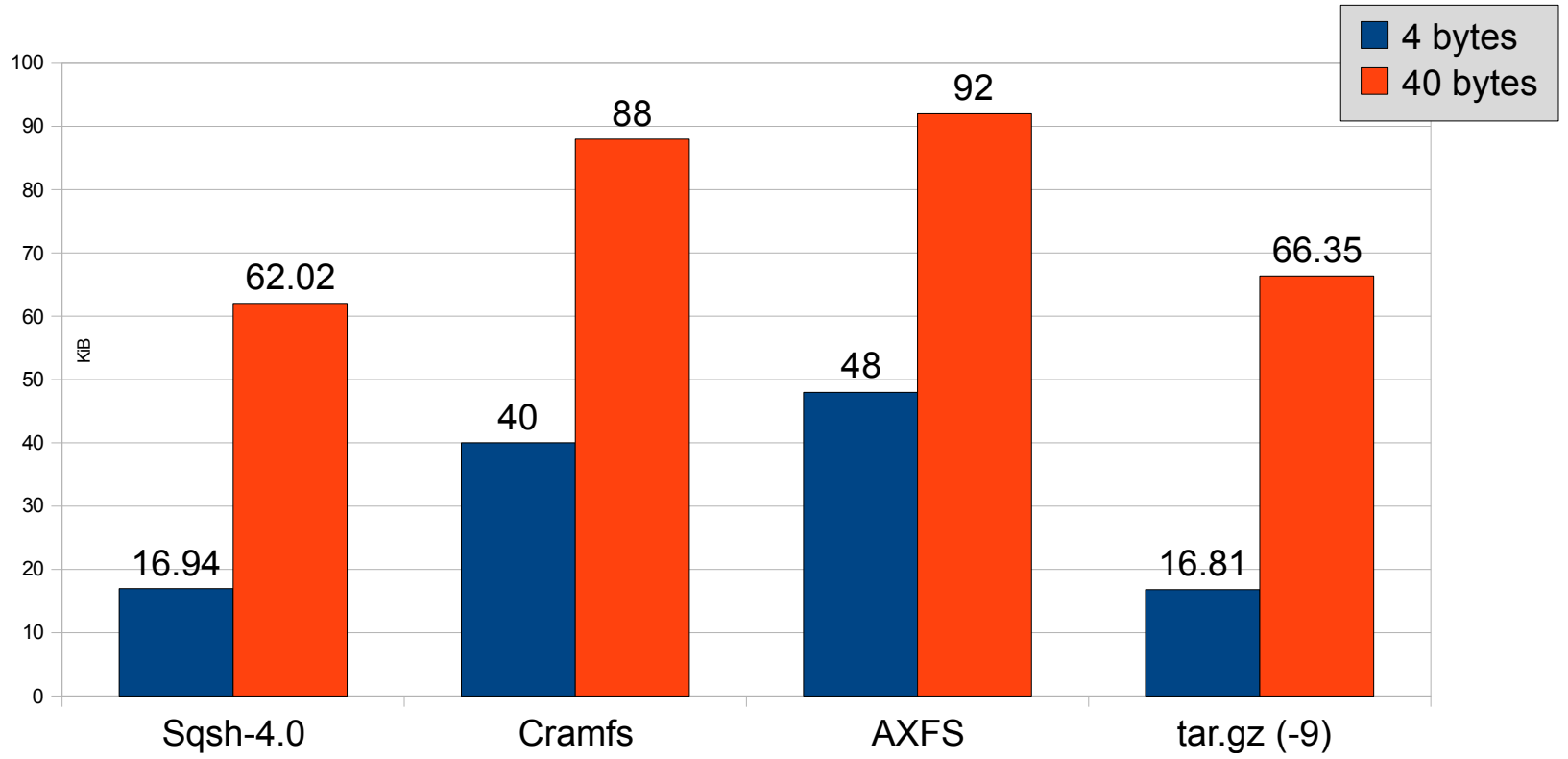
released in Squashfs 3.3

4.0 to be released soon...

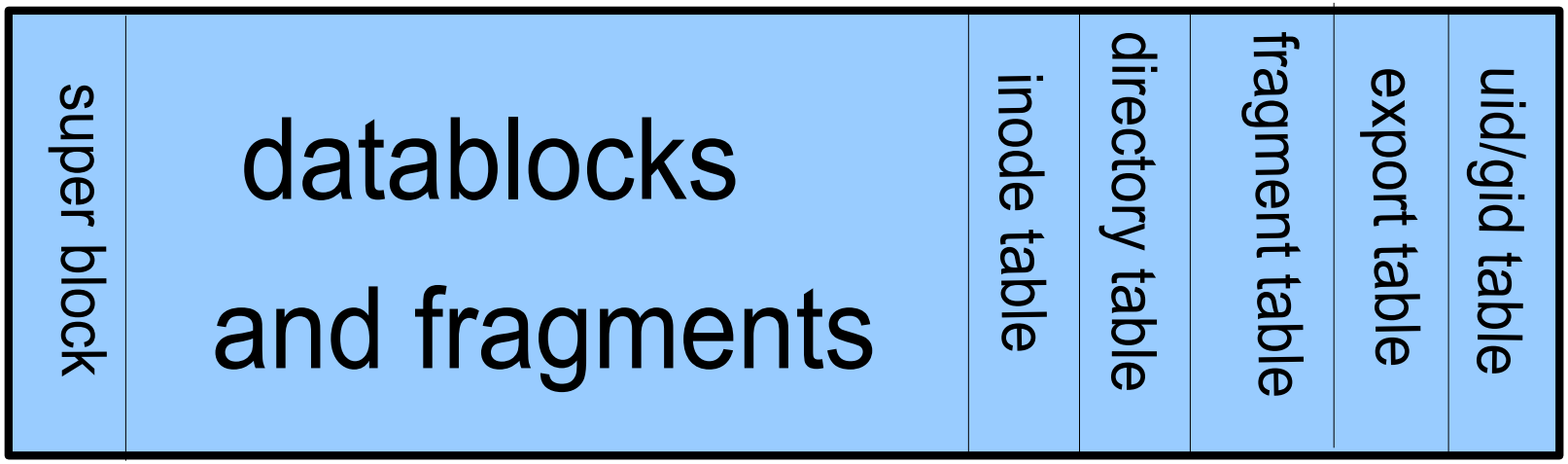
Filesystem comparison (linux 2.6 source tree)



Filesystem comparison (1200 very small files)



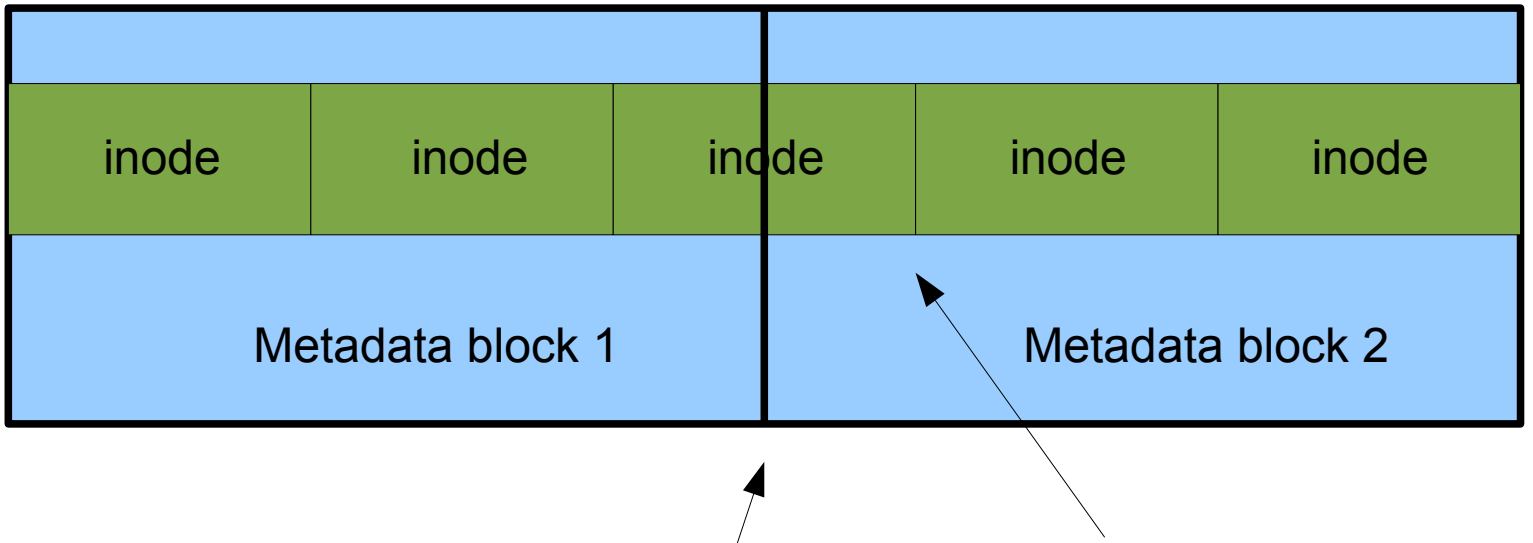
SquashFS layout overview



Filesystem layout

Packed metadata blocks in inode and directory tables

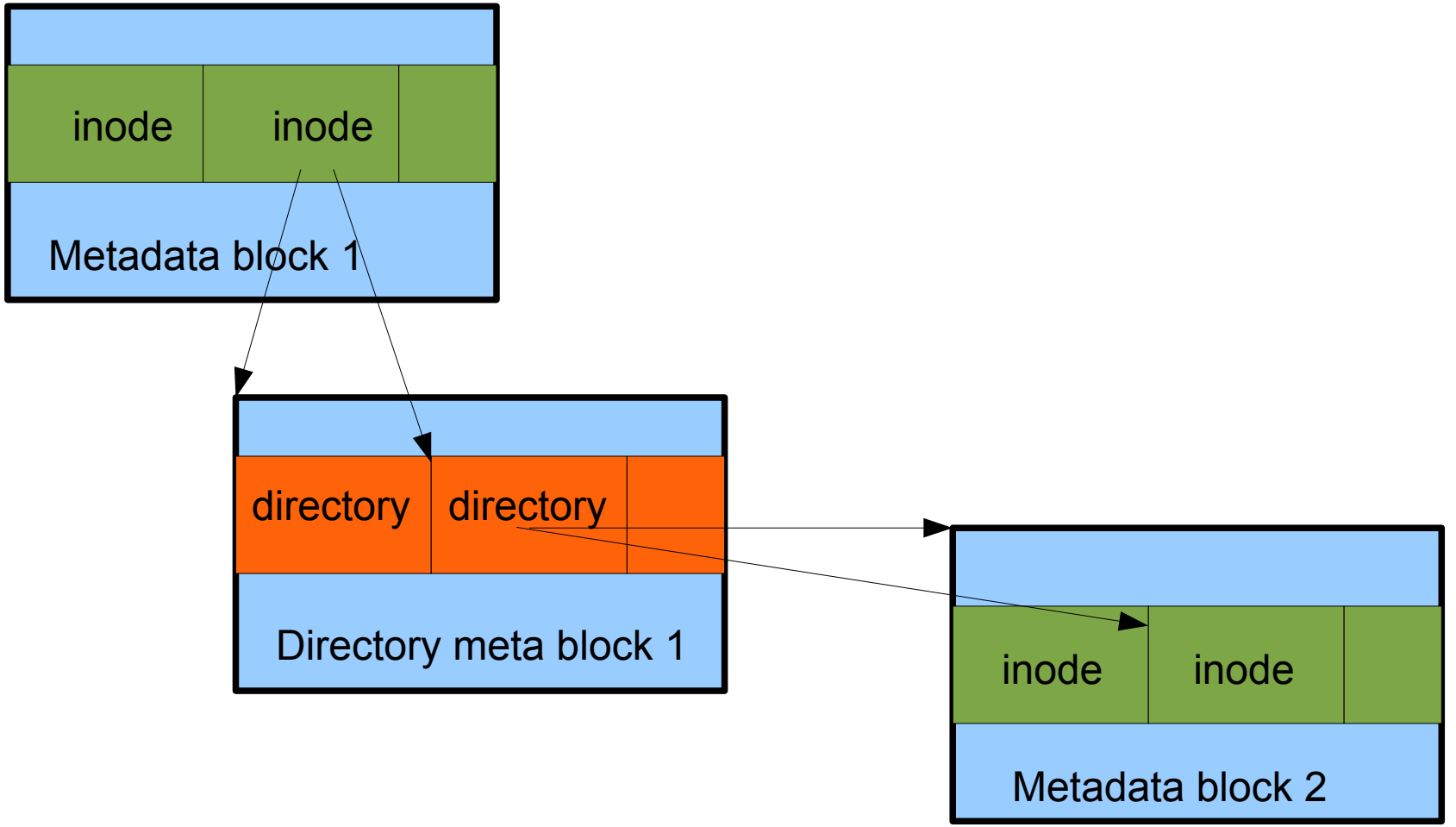
SquashFS
layout overview (cont)



Inode location = Start-block + offset

Likewise Directory location = Start-block + offset

SquashFS layout overview (cont)

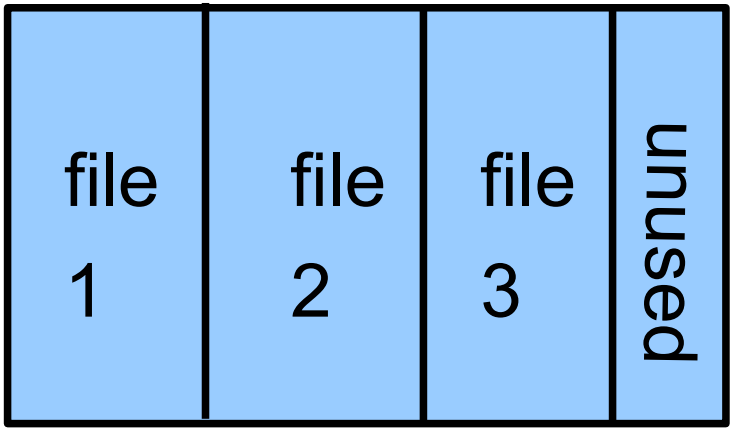


Fragments

- Introduced in release 2.0
- Many files are smaller than the block size, especially using large blocks (128 KiB)
  - Loss of compression
    - compress block size tends to average size of file
    - No point using large blocks

Fragments (cont)

- Pack files smaller than block size into shared blocks and compress in one whole



Fragments (cont)

- Fragments exhibit “locality of reference”
  - Best compression achieved by packing together similar files
    - Good approximation, sort directories and pack in alphabetical order
  - “Best-fit” strategy gets worse compression
  - Packing tail-ends of files larger than datablock also reduces compression
    - Mksquashfs -always-use-fragments

Directory indexes

- Introduced in release 2.1
  - Original directories limited to 512 KiB
    - A company wanted to use SquashFS to store 418,797+ files in one directory
    - Without directory indexes ls took 16 hours! (slow h/w).
- Fast dentry operations on large directories (larger than 64 KiB)
  - Only one metadata block decompression irrespective of directory size

Mainline kernel integration

- First attempt on 14th March 2005
- Initially went well. However attempt stalled on a number of key issues
  - 4 GiB filesystem and file limitation
  - No fixed endian layout
    - Complex packed bit-field macros need to swap between different endianness
  - No ".." and "." names returned by readdir
- Quote "So we are replacing severely-limited cramfs with also limited Squashfs"

Kernel integration (cont)

- Encouraged new 3.0 layout
  - Directly addressed criticisms in mainline attempt
  - Greater than 4 GiB filesystems and files
  - “.” and “..”, with real inode numbers
  - Hardlinks
  - NFS exporting
- SquashFS 3.0 a more “grown-up” filesystem

Kernel integration (cont)

- Squashfs 3.0 released 15th March 2006
  - About a year since first mainline attempt
- Still with bit-fields and both big-little endian layouts, why?
  - Released to show progress was being made
  - Companies hitting limits in 2.x and asking for the new layout
  - Bit-field removal and fixed little-endian a lot of work, it would delay 3.0 by at least 6 months

New 4.0 layout

- CE Linux Forum (CELF)
  - Offered to contribute financially to another mainline attempt earlier this year
- New 4.0 layout
  - Finally got rid of bit fields
  - Moved to fixed little endian
  - Couple of other filesystem tweaks
    - Increased max uids/gids to 65536. Merged compressed uid/gid table

New 4.0 layout

- Primary objective, no loss of compression compared to 3.0 layout
  - Careful re-ordering of filesystem metadata to maximise compression

	3.4	4	%
Reg	32	32	0
Xreg	40	56	40
Dir	28	32	14
Xdir	31	40	29
Sym	18	24	33
Dev	18	24	33

New 4.0 layout (cont)

- New patches submitted to LKML on 17th October 2008
  - Generally favourable comments
- New patches following comments submitted to LKML on 28th October 2008
- Third set of patches necessary following second set of comments
  - Hopefully will go into linux-next and see mainline in 2.6.29

Improving compression

- Most people just type mksquashfs
  - Default options optimised for normal usage
- Increase block size
  - Maximum compression now 1 MiB
- Use -alway-use-fragments option
  - Pack large file 'tail ends' rather than just pack small files
- Use -sort option
  - Sort similar files so that they're placed in the same compressed fragment

Improving memory usage

- Squashfs caches the last couple of fragments read
  - block_size * number_of_cached_fragments (default 3)
- Kernel configuration
  - Select SQUASHFS_EMBEDDED
  - Then change option SQUASHFS_FRAGMENT_CACHE_SIZE
- Use -no-fragments option on Mksquashfs
- Use smaller blocks

To Do

- Obviously get into mainline
- Xattr and ACL support
  - Last non-supported 'big filesystem' feature
- Back to basics – improve compression...

Frequently asked questions

- Why doesn't Squashfs officially support lzma compression?
  - Lzma not part of mainline kernel
  - Will lzma ever become mainline?
    - Probably needs clean-up to be accepted
  - Adopting lzma officially will likely block Squashfs until lzma becomes accepted
- New 4.0 layout includes compression field in superblock
  - Should simplify third-party lzma patches

Questions?