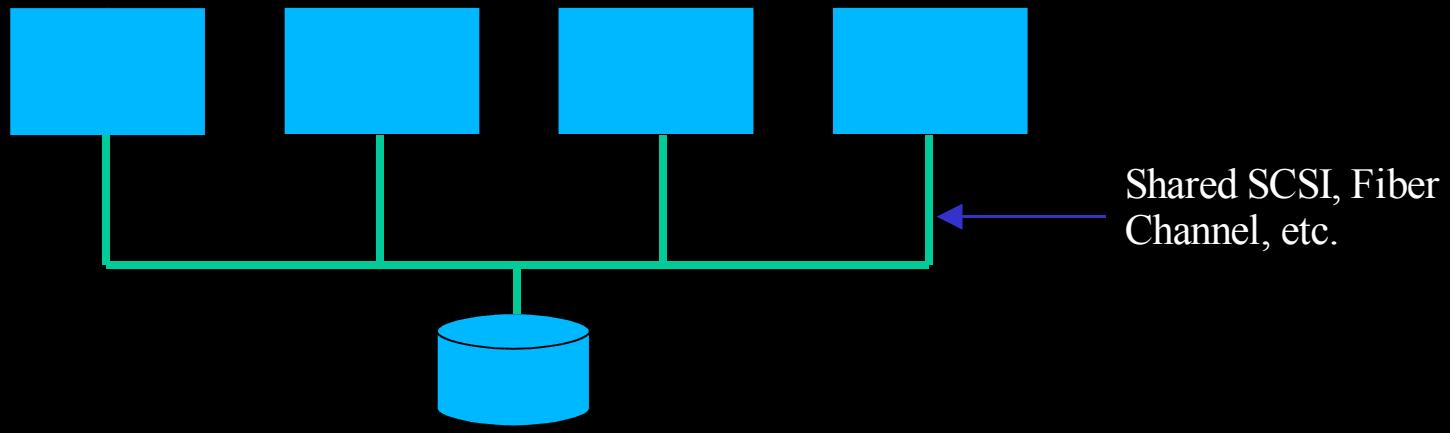
Oracle Cluster File System on Linux Version 2

Kurt Hackel
Señor Software Developer
Oracle Corporation

ORACLE

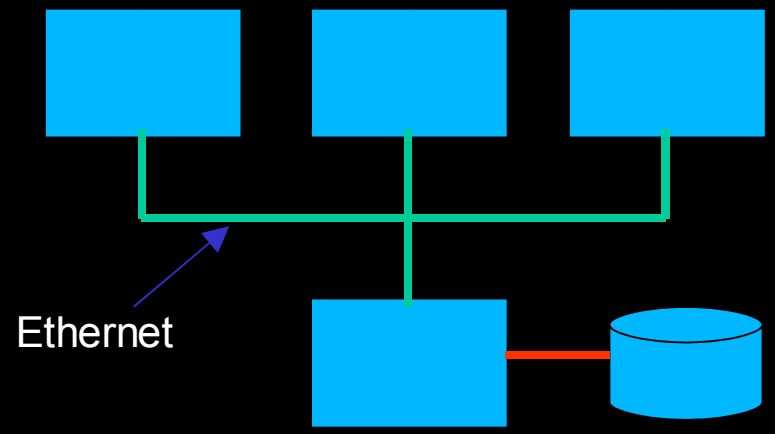
What is OCFS?

- GPL'd Extent Based Cluster File System
- Is a shared disk clustered file system
- Allows two or more nodes to access the same file system
- File system is mounted natively on all nodes
- Supports a maximum of 32 nodes



Is it like NFS?

- No.
- In NFS, the file system is hosted by one node
- Rest of the nodes access the file system via the network
- Single point of failure
- No node recovery
- Slow data throughput



Why does Oracle need it?

- Oracle's Real Application Cluster (RAC) database, uses a shared disk
- As most OSes do not provide a shared disk cluster file system, RAC data files, control files, etc. need to exist on a raw partition
- Raw is hard to manage
- Moreover, Linux 2.4 allows a max of 255 raw partitions
- No auto-extending of partitions

Why does Oracle need it?

- OCFS allows for easier management as it looks and feels just like a regular file system
- No limit on number of files
- Allows for very large files (max 2TB)
- Max volume size 32G (4K block) to 8T (1M block)
- Oracle DB performance is comparable to raw

What does the database provide?

- Multi-node data caching
- Multi-node data locking
- Journals it's own operations (DB logfiles)

What's new in OCFS2?

- Support shared ORACLE_HOME
  - Improved meta data caching
  - Improved meta data journalling
  - Improved node recovery
  - Faster data allocation
  - Context Dependant Symbolic Links (CDSL)
  - Cleaner code

How do I use it?

- Hardware Setup
  - 2+ node setup with some sort of shared disk
  - Shared disk could be Shared SCSI, Fiber Channel, etc.
  - For testing purposes, recommend using FireWire (very cheap)
  - http://oss.oracle.com/projects/firewire
  - OSS site has information and kernels with FireWire fixes

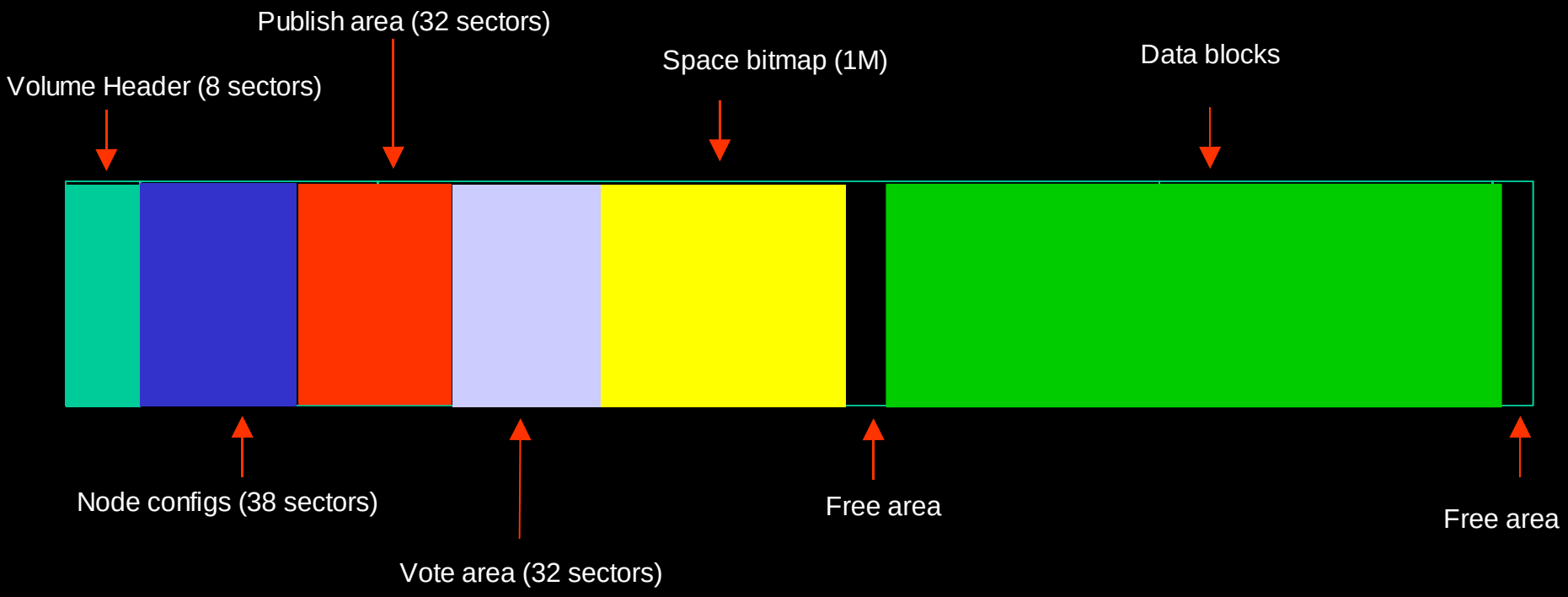
Process Architecture

- OCFS is a kernel module
- On the first mount creates 3 kernel threads
  - [ocfs2nm-N] => one for each mounted volume. Thread runs in a loop reading the volume for any lock requests from other nodes.
  - [ocfs2cmt-N] => journal checkpointing thread. Commits pending transactions and releases related locks.
  - [ocfs2lsnr] => one on a node. Is a listener for the network dlm.

Process Architecture

- The third important pid is that of the user-space process which is accessing the fs. e.g., cp, mv, dbwr, etc.
- All lock requests on a node are triggered by the user-space process.
- All lock requests by other nodes are serviced by the ocfs2nm-N thread, using kernel worker threads.
- Recovery threads come and go as nodes die.

Volume Layout



Note: Not drawn to scale

Node Configuration

- Node name, ip address, ip port and guid is stored in this area
- Slots 0 to 31 represent node numbers 0 - 31
- Node number is auto-allocated the first time a node mounts a volume
- A node could have different node numbers across multiple ocfs volumes
- /proc/ocfs2/<volume_num>/nodenum
- OCFS identifies a node by its guid

Publish Area

- Every node owns one sector for writing, aka, its publish sector
- In it, the nodes write the timestamp at regular intervals to indicate to the other nodes that they are alive
- Nodes also use their publish sector to request locks on a resource
- Resources are structures on disk and its number is its byte offset

Vote Area

- Every node owns one sector for writing, aka, its vote sector
- In it, nodes vote for the resource lock asked to by another node
- Requesting node collects the votes from all the nodes and takes the lock if all vote OK
- The lock state is written on the disk (for files in the file entry, for bitmap in the bitmap lock sector)

Distributed Lock Manager

- Network DLM is strongly preferred method
- OCFS requires locks only for the file system meta-data changes
- Does not protect file data changes
- Expects the application to be cluster-aware
- Oracle RAC is cluster-aware and it performs its own intelligent caching and locking of file data

Distributed Lock Manager

- Network-based dlm functions similarly.
- In it, the node requesting a vote just sends a vote-request packet to all interested nodes
- The nodes inturn reply using the vote-reply packet
- When activated, the publish sector is only used to identify alive nodes (heartbeat) whereas the vote sector is unused
- The disk-based dlm gets automatically activated whenever one or more “alive” nodes is not heard of on the network

Space Management - Bitmap

- Each bit in the space bitmap indicates free/alloc state of a data block
- Bitmap size is fixed to 1M
- Size of block size determines max size of volume

  max_vol_size = block_size * 1M * 8

- Block sizes can be 4K, 8K, 32K, 64K, 128K, 256K, 512K or 1M

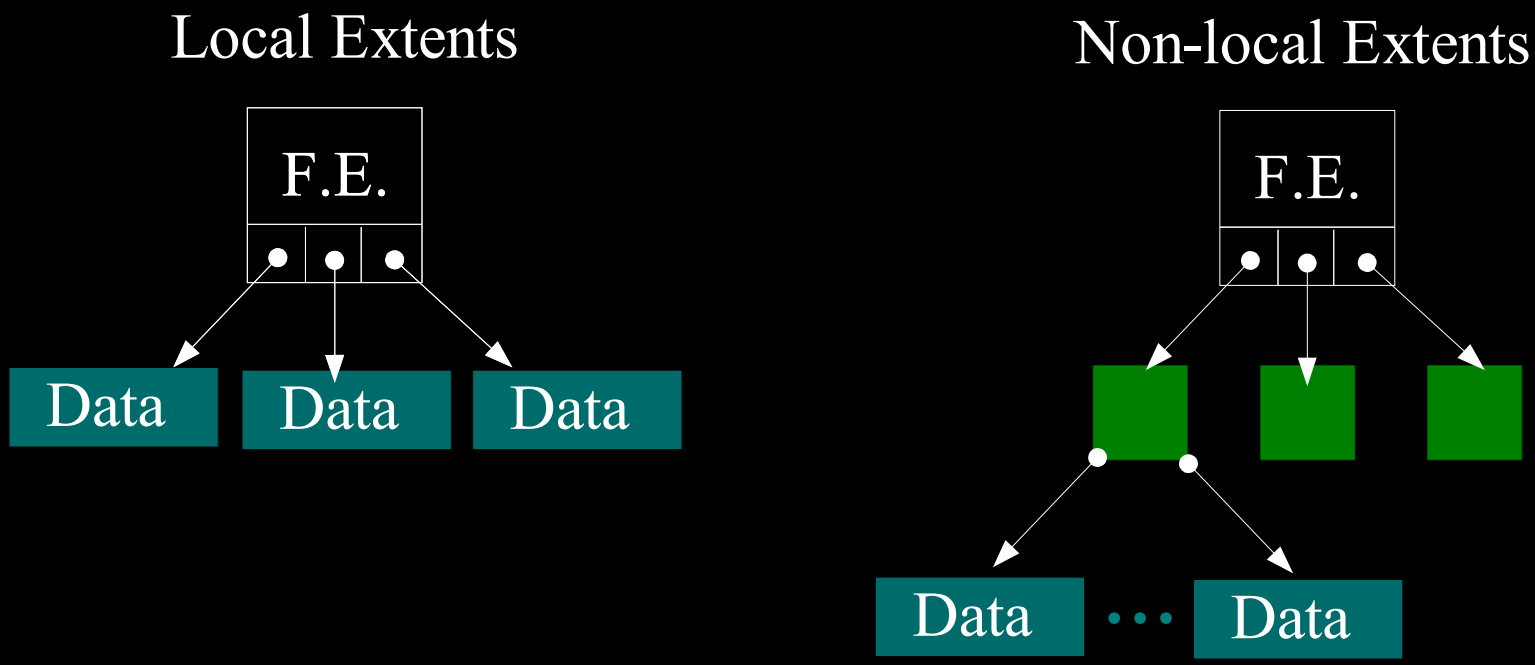
Space Management

- File data is allocated space from the same bitmap
- Each meta-data on disk has a lock structure which holds the lock state
- System files allocated using the same scheme
- System files are used to allocate metadata, store the journal, etc.
- Are hidden for regular file system calls

Space Management – File

- Uses extent based space allocation for files rather than the block based (ext2)
- Requires less accounting for very large files
- File entry initially has 3 direct extent pointers
- When file has >3 extents, the extent pointers become indirects
- When file has >54 extents, the extent pointers become double indirects

Space Management – File



- Green squares are indirect blocks which hold 18 extent pointers each.
- Can have up to three levels of indirect pointers before you've run out of theoretical space.

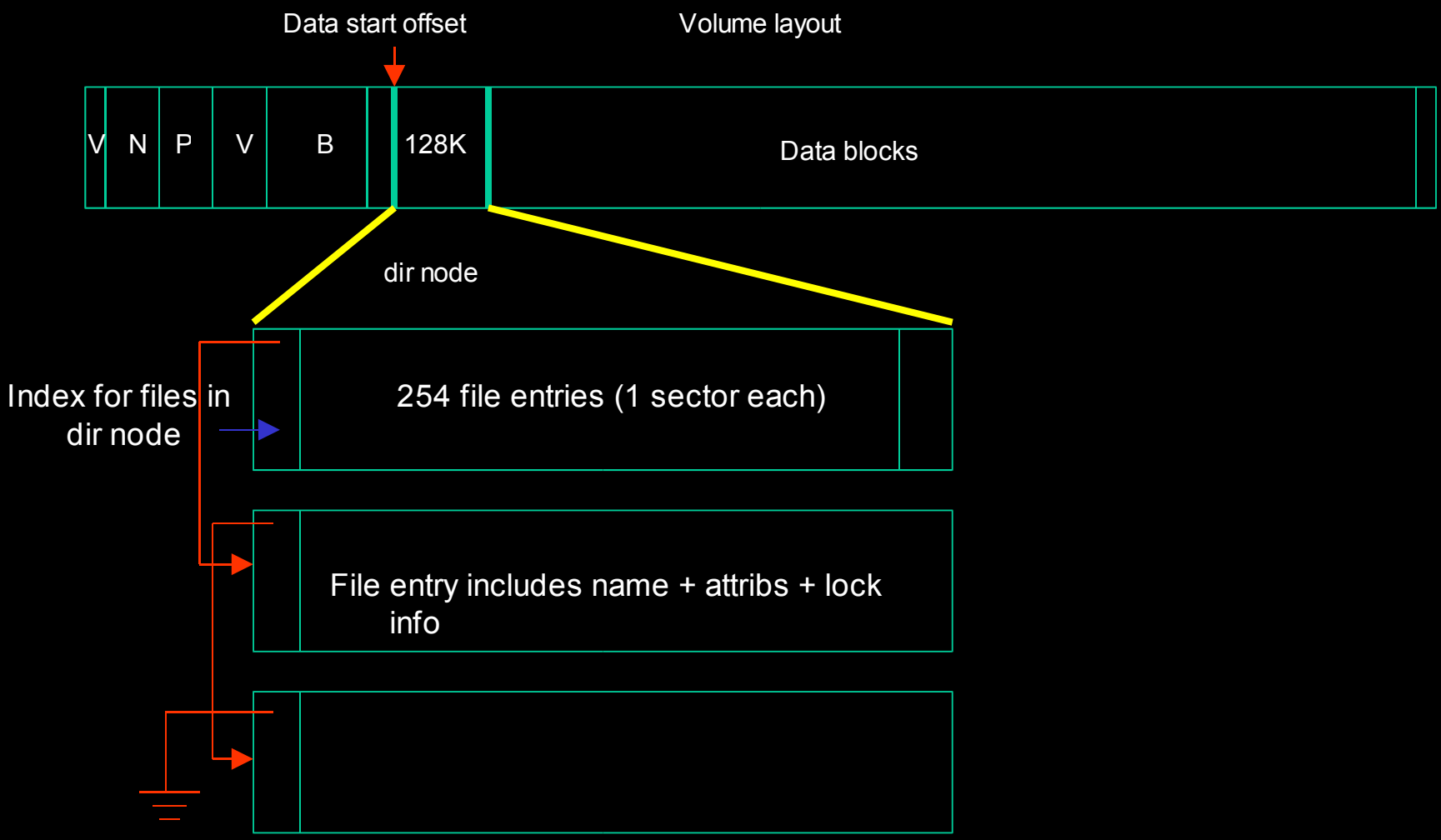
Space Management – Local Alloc

- One “window” per node, only use local alloc on smaller space allocations. Reduces lock contention on main bitmap.
- All bits in window are set on main bitmap, local alloc starts clean.
- As space is used, local alloc bits are set.
- Unused bits are returned to bitmap on shutdown, recovery, or on a window move.

Space Management – Directory

- Directory is a 128K block
- It includes 254 (512 byte) file entries
- Each file entry represents a file, sub-dir or link
- File entry houses the name of the file/sub-dir/link, attributes, locking info
- When the number of file in a dir > 254, another 128K block is linked

Space Management – Directory



Journalling

- One 8 MB journal file per node
- Block based journalling using the same JBD subsystem as Ext3
- JBD keeps track of changed blocks and writes them to the journal before flushing them out to disk
- OCFS2 retains locks on journalled objects until they are flushed by ocfs2-cmt thread (on demand or on timeout)

Recovery

- NM thread detects node death, launches recovery thread
- Locks owned by dead node cannot be retaken until it has been recovered.
- Recovery Steps:
  - Lock journal file
  - If node needs recovery, replay journal and recover local alloc.
  - Mark node clean
  - Unlock journal file

Caching

- Cache Meta Data using “Sequence Numbers”
  - In buffer_head private bits
  - In inode private data
  - Global sequence number
- On block reads, BH sequence # is compared with inode.
- Increment the inode sequence # when another node locks it.
- Global incremented with each inode and new sequence #'s are set from it.

CDSL - "Context Dependant Symbolic Links"

- Allows node specific files on OCFS2
- Use symlink mechanism to indicate a CDSL file
- Link is followed to a CDSL directory using substitution of hostname
- All tools for creating/modifying CDSL are entirely within userspace and require no kernel hooks
- Very beta feature at the moment

Improvements in Linux

- Make VFS cluster-aware
- Extend locks in VFS to cluster-wide locks
- Generic DLM services
- Cluster Manager
- IO Fencing
- JBD Improvements

ORACLE

http://oss.oracle.com/projects/ocfs2/

Oracle on Linux File Sizes

	32bit	64bit
Oracle 9i tablespace		
Can have up to 1022 files per tablespace with 4 Million blocks in each file		
Max blocksize	16k	32k
Max Filesize	64GB	128GB
Oracle 10g smallfile tablespace		
Can have up to 1022 files per tablespace with 4 Million blocks in each file		
Max blocksize	16k	32k
Max Filesize	64GB	128GB
Oracle 10g largefile tablespace		
Has only 1 large file with up to 4 Billion blocks		
Max blocksize	16k	32k
Max Filesize	64TB **	64TB **
** Max Filesize limited to max volume size determined by the kernel		
2.4 linux kernel		
Max volume size	2TB	2TB
Max raw devices	255	255
2.6 Linux kernel		
Max volume size	16TB	peta/exa B's
**OCFS V1 & V2 ** Both use 64 bit addressing **		
Default blocksize	128k	128k
Block sizes	4k, 8k, 32k, 64k, 128k, 256k, 512k or 1MB	

max_vol_size = block_size * 1M * 8

max ocfs vol size is limited to the max volume size determined by the kernel