Divide and conquer – Shared disk cluster file systems shipped with the Linux kernel

Udo Seidel

Shared file systems

- Multiple server access same data
- Different approaches
  - Network based, e.g. NFS, CIFS
  - Clustered
    - Shared disk, e.g. CXFS, CFS, GFS(2), OCFS2
    - Distributed parallel, e.g. Lustre, Ceph

History

- GFS(2)
  - First version in the mid 90’s
  - Started on IRIX, later ported to Linux
  - Commercial background: Sistina and RedHat
  - Part of Vanilla Linux kernel since 2.6.19
- OCFS2
  - OCFS1 for database files only
  - First version in 2005
  - Part of Vanilla Linux kernel since 2.6.16

Features/Challenges/More

- As much as possible similar to local file systems
  - Internal setup
  - management
- Cluster awareness
  - Data integrity
  - Allocation

Framework

- Bridges the gap between one-node and cluster
- 3 main components
  - Cluster-ware
  - Locking
  - Fencing

Framework GFS2 (I)

- Cluster-ware of general purpose
  - More flexible
  - More options/functions
  - More complexity
  - Configuration files in XML
- Locking uses cluster framework too
- `system-config-cluster` OR Conga OR `vi` & `scp`

Framework GFS2 (II)

```
# cat /etc/cluster/cluster.conf
<?xml version="1.0" ?>
<cluster config_version="3" name="gfs2">
<fence_daemon post_fail_delay="0" post_join_delay="3"/>
<clusternodes>
<clusternode name="node0" nodeid="1" votes="1">
<fence/>
</clusternode>
<clusternode name="node1" nodeid="2" votes="1">
...
</cluster>
#
```

Framework OCFS2 (I)

- Cluster-ware just for OCFS2
  - Less flexible
  - Less options/functions
  - Less complexity
  - Configuration file in ASCII
- Locking uses cluster framework too
- `ocfs2console` OR `vi` & `scp`

Framework OCFS2 (II)

```
# cat /etc/ocfs2/cluster.conf
node:
        ip_port = 7777
        ip_address = 192.168.0.1
        number = 0
        name = node0
        cluster = ocfs2
...
cluster:
        node_count = 2

#
```

Locking

- Distributed Lock Manager (DLM)
- Based on VMS-DLM
- Lock modes
  - Exclusive Lock (EX)
  - Protected Read (PR)
  - No Lock (NL)
  - Concurrent Write Lock (CW) – GFS2 only
  - Concurrent Read Lock (CR) – GFS2 only
  - Protected Write (PR) – GFS2 only

Locking - Compatibility

	Existing Lock					
Requested Lock	**NL**	**CR**	**CW**	**PR**	**PW**	**EX**
NL	Yes	Yes	Yes	Yes	Yes	Yes
CR	Yes	Yes	Yes	Yes	Yes	No
CW	Yes	Yes	Yes	No	No	No
PR	Yes	Yes	No	Yes	No	No
PW	Yes	Yes	No	No	No	No
EX	Yes	No	No	No	No	No

Fencing

- Separation of host and storage
  - Power Fencing
    - Power switch, e.g. APC
    - Server side, e.g. IPMI, iLO
    - Useful in other scenarios
    - Post-mortem more difficult
  - I/O fencing
    - SAN switch, e.g. Brocade, Qlogic
    - Possible to investigate “unhealthy” server

Fencing - GFS2

- Both fencing methods
- Part of cluster configuration
- Cascading possible

Fencing - OCFS2

- Only power fencing
  - Only self fencing

GFS2 – Internals (I)

- Superblock
  - Starts at block 128
  - Expected data + cluster information
  - Pointers to master and root directory
- Resource groups
  - Comparable to cylinder groups of traditional Unix file system
  - Allocatable from different cluster nodes -> locking granularity

GFS2 – Internals (II)

- Master directory
  - Contains meta-data, e.g journal index, quota, ...
  - Not visible for `ls` and Co.
  - File system unique and cluster node specific files
- Journaling file system
  - One journal per cluster node
  - Each journal accessible by all nodes (recovery)

GFS2 – Internals (III)

- Inode/Dinode
  - Usual information, e.g. owner, mode, time stamp
  - Pointers to blocks: either data or pointer
  - Only one level of indirection
  - “stuffing”
- Directory management via Extendible Hashing
- Meta file statfs
  - `statfs()`
  - Tuning via `sysfs`

GFS2 – Internals (IV)

- Meta files
  - `jindex` directory containing the journals
    - `journalX`
  - `rindex` Resource group index
  - `quota`
  - `per_node` directory containing node specific files

GFS2 – what else

- Extended attributes `xattr`
- ACL’s
- Local mode = one node access

OCFS2 – Internals (I)

- Superblock
  - Starts at block 3 (1+2 for OCFS1)
  - Expected data + cluster information
  - Pointers to master and root directory
  - Up to 6 backups
    - at pre-defined offset
    - at 2^n Gbyte, n=0,2,4,6,8,10
- Cluster groups
  - Comparable to cylinder groups of traditional Unix file system

OCFS2 – Internals (II)

- Master or system directory
  - Contains meta-data, e.g journal index, quota, ...
  - Not visible for `ls` and Co.
  - File system unique and cluster node specific files
- Journaling file system
  - One journal per cluster node
  - Each journal accessible by all nodes (recovery)

OCFS2 – Internals (III)

- Inode
  - Usual information, e.g. owner, mode, time stamp
  - Pointers to blocks: either data or pointer
  - Only one level of indirection
- `global_inode_alloc`
  - Global meta data file
  - `inode_alloc` node specific counterpart
- `slot_map`
  - Global meta data file
  - Active cluster nodes

OCFS2 – Internals (IV)

- `orphan_dir`
  - Local meta data file
  - Cluster aware deletion of files in use
- `truncate_log`
  - Local meta data file
  - Deletion cache

OCFS2 – what else

- Two versions: 1.2 and 1.4
  - Mount compatible
  - Framework not network compatible
  - New features disabled per default
- For 1.4:
  - Extended attributes `xattr`
  - Inode based snapshotting
  - preallocation

File system management

- Known/expected tools + cluster details
  - `mkfs`
  - `mount/umount`
  - `fsck`
- File system specific tools
  - `gfs2_XXXX`
  - `tunefs.ocfs2, debugfs.ocfs2`

GFS2 management (I)

- File system creation needs additional information
  - Cluster name
  - Unique file system identifier (string)
  - Optional:
    - Locking mode to be used
    - number of journals
  - Tuning by changing default size for journals, resource groups, ...

GFS2 management (II)

- Mount/umount
  - No real syntax surprise
  - First node checks all journals
  - Enabling ACL, quota, single node mode

GFS2 management (III)

- File system check
  - Journal recovery of node X by node Y
  - Done by one node
  - file system offline anywhere else
  - Known phases
    - Journals
    - Meta data
    - References: data blocks, inodes

GFS2 tuning (I)

- `gfs2_tool`
  - Most powerful
    - Display superblock
    - Change superblock settings (locking mode, cluster name)
    - List meta data
    - freeze/unfreeze file system
    - Special attributes, e.g. appendonly, noatime
  - Requires file system online (mostly)

GFS2 tuning (II)

- `gfs2_edit`
  - Logical extension of `gfs2_tool`
  - More details, e.g. node-specific meta data, block level
- `gfs2_jadd`
  - Different sizes possible
  - No deletion possible
  - Can cause data space shortage

GFS2 tuning (III)

- `gfs2_grow`
  - Needs space in meta directory
  - Online only
  - No shrinking

OCFS2 management (I)

- File system creation
  - no additional information needed
  - Tuning by optional parameters
- Mount/umount
  - No real syntax surprise
  - First node checks all journals
  - Enabling ACL, quota, single node mode

OCFS2 management (II)

- File system check
  - Journal recovery of node X by node Y
  - Done by one node
  - file system offline anywhere else
  - Fixed offset of superblock backup handy
  - Known phases
    - Journals
    - Meta data
    - References: data blocks, inodes

OCFS2 tuning (I)

- `tunefs.ocfs2`
  - Display/change file system label
  - Display/change number of journals
  - Change journal setup, e.g.size
  - Grow file system (no shrinking)
  - Create backup of superblock
  - Display/enable/disable specific file system features
    - Sparse files
    - “stuffed” inodes

OCFS2 tuning (II)

- `debugs.ocfs2`
  - Display file system settings, e.g. superblock
  - Display inode information
  - Access meta data files

Volume manager

- Necessary to handle more than one LUN/partition
- Cluster-aware
- Bridge feature gap, e.g. volume based snapshotting
- CLVM
- EVMS – OCFS2 only

Key data - comparison

	GFS2	OCFS2
Maximum # of cluster nodes	Supported 16 (theoretical: 256)	256
journaling	Yes	Yes
Cluster-less/local mode	Yes	Yes
Maximum file system size	25 TB (theoretical: 8 EB)	16 TB (theoretical: 4 EB)
Maximum file size	25 TB (theoretical: 8 EB)	16 TB (theoretical: 4 EB)
POSIX ACL	Yes	Yes
Grow-able	Yes/online only	Yes/online and offline
Shrinkable	No	No
Quota	Yes	Yes
O_DIRECT	On file level	Yes
Extended attributes	Yes	Yes
Maximum file name length	255	255
File system snapshots	No	No

Summary

- GFS2 longer history than OCFS2
- OCFS2 setup simpler and easier to maintain
- GFS2 setup more flexible and powerful
- OCFS2 getting close to GFS2
- Dependence on choice of Linux vendor

References

http://sourceware.org/cluster/gfs/

http://www.redhat.com/gfs/

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

http://sources.redhat.com/cluster/wiki/

http://sourceware.org/lvm2/

http://evms.sourceforge.net/

Thank you!