THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
EUROPE

Comparison of FOSS distributed filesystems

Marian *“HackMan”* Marinov, SiteGround

THE LINUX FOUNDATION

Who am I?

- Chief System Architect of SiteGround
- Sysadmin since 96
- Teaching Network Security & Linux System Administration in Sofia University
- Organizing biggest FOSS conference in Bulgaria - OpenFest

Why were we considering shared FS?

- We are a hosting provider
- We needed shared filesystem between VMs and containers

- Different size of directories and relatively small files
- Sometimes milions of files in a single dir

The story of our storage endeavors

- We started with DBRD + OCFS2
- I did a small test of cLVM + ATAoE
- We tried DRBD + OCFS2 for MySQL clusters
- We then switched to GlusterFS
- Later moved to CephFS
- and finally settled on good old NFS :)
  but with the storage on Ceph RBD

Which filesystems were tested

- CephFS
- GlusterFS
- MooseFS
- OrangeFS
- BeeGFS - no stats

I haven't played with Lustre or AFS
And because of our history with OCFS2 and GFS2 I skipped them

Test cluster

- CPU i7-3770 @ 3.40GHz / 16G DDR3 1333MHz
- SSD SAMSUNG PM863
- 10Gbps Intel x520
- 40Gbps Qlogic QDR Infiniband
- IBM Blade 10Gbps switch
- Mellanox Infiniscale-IV switch

What did I test?

- Setup
- Fault tollerance
- Resource requirements (client & server)
- Capabilities (Redundancy, RDMA, caching)
- FS performance (creation, deletion, random read, random write)

Complexity of the setup

- CephFS - requires a lot of knowledge and time
- GlusterFS - relatively easy to setup and install
- OrangeFS - extremely easy to do basic setup
- MooseFS - extremely easy to do basic setup
- DRBD + NFS - very complex setup if you want HA
- BeeGFS -

CephFS

- Fault tollerant by desing...
  - until all MDSes start crashing
    - single directory may crash all MDSes
    - MDS behind on trimming
    - Client failing to respond to cache pressure
- Ceph has redundancy but lacks RDMA support

CephFS

- Uses a lot of memory on the MDS nodes
- Not suitable to run on the same machines as the compute nodes
- Small number of nodes 3-5 is a no go

CephFS tunning

- Placement Groups(PG)
- mds_log_max_expiring & mds_log_max_segments fixes the problem with trimming
- When you have a lot of inodes, increasing mds_cache_size works but increases the memory usage

CephFS fixes

- Client XXX failing to respond to cache pressure
  - This issue is due to the current working set size of the inodes.

```
ceph daemon mds.$(hostname) perf dump | grep inode
```

```
"inode_max": 100000, - max cache size value
"inodes": 109488, - currently used
```

The fix for this is setting the mds_cache_size in /etc/ceph/ceph.conf accordingly.

CephFS fixes

- Client XXX failing to respond to cache pressure
  - Another “FIX” for the same issue is to login to current active MDS and run:

```
/etc/init.d/ceph restart mds
```

  - This way it will change currnet active MDS to another server and will drop inode usage

GlusterFS

- Fault tollerance
  - in case of network hikups sometimes the mount may become inaccesable and requires manual remount
  - in case one storage node dies, you have to manually remount if you don't have local copy of the data
- Capabilities - local caching, RDMA and data Redundancy are supported. Offers different ways for data redundancy and sharding

GlusterFS

- High CPU usage for heavily used file systems
  - it required a copy of the data on all nodes
  - the FUSE driver has a limit of the number of small operations, that it can perform
- Unfortunatelly this limit was very easy to be reached by our customers

GlusterFS Tunning

- use distributed and replicated volumes instead of only replicated
  - gluster volume create VOL replica 2 stripe 2 ....
- setup performance parameters

GlusterFS Tunning

```
volume set VOLNAME performance.cache-refresh-timeout 3
volume set VOLNAME performance.io-thread-count 8
volume set VOLNAME performance.cache-size 256MB
volume set VOLNAME performance.cache-max-file-size 300KB
volume set VOLNAME performance.cache-min-file-size 0B
volume set VOLNAME performance.readdir-ahead on
volume set VOLNAME performance.write-behind-window-size 100KB
```

GlusterFS Tunning

```
volume set VOLNAME features.lock-heal on
volume set VOLNAME cluster.self-heal-daemon enable

volume set VOLNAME cluster.metadata-self-heal on

volume set VOLNAME cluster.consistent-metadata on

volume set VOLNAME cluster.stripe-block-size 100KB

volume set VOLNAME nfs.disable on
```

FUSE tunning

FUSE	GlusterFS
entry_timeout	entry-timeout
negative_timeout	negative-timeout
attr_timeout	attribute-timeout

mount eveyrhting with “-o intr” :)

MooseFS

- Reliability with multiple masters
- Multiple metadata servers
- Multiple chunkservers
- Flexible replication (per directory)
- A lot of stats and a web interface

BeeGFS

- Metadata and Storage nodes are replicated by design
- FUSE based

OrangeFS

- No native redundancy uses corosync/pacemaker for HA
- Adding new storage servers requires stopping of the whole cluster
- It was very easy to break it

Ceph RBD + NFS

- the main tunning goes to NFS, by using the cachefilesd
- it is very important to have cache for both accessed and missing files
- enable FS-Cache by using "-o fsc" mount option

Ceph RBD + NFS

- verify that your mounts are with enabled cache:

```
# cat /proc/fs/nfsfs/volumes
NV SERVER  PORT DEV     FSID                                FSC
v4 aaaaaaaa  801     0:35     17454aa0fddaa6a5:96d7706699eb981b yes
v4 aaaaaaaa  801     0:374   7d581aa468faac13:92e653953087a8a4 yes
```

Sysctl tunning

```
fs.nfs.idmap_cache_timeout = 600
fs.nfs.nfs_congestion_kb = 127360
fs.nfs.nfs_mountpoint_timeout = 500
fs.nfs.nlm_grace_period = 10
fs.nfs.nlm_timeout = 10
```

Sysctl tunning

```
# Tune the network card polling
net.core.netdev_budget=900
net.core.netdev_budget_usecs=1000
net.core.netdev_max_backlog=300000
```

Sysctl tunning

```
# Increase network stack memory
net.core.rmem_max=16777216
net.core.wmem_max=16777216
net.core.rmem_default=16777216
net.core.wmem_default=16777216
net.core.optmem_max=16777216
```

Sysctl tunning

```
# memory allocation min/pressure/max.
# read buffer, write buffer, and buffer space
net.ipv4.tcp_rmem = 4096 87380 134217728
net.ipv4.tcp_wmem = 4096 65536 134217728
```

Sysctl tunning

```
# turn off selective ACK and timestamps
net.ipv4.tcp_sack = 0
net.ipv4.tcp_timestamps = 0
net.ipv4.tcp_low_latency = 1
# scalable or bbr
net.ipv4.tcp_congestion_control = scalable
```

Sysctl tunning

```
# Increase system IP port range to allow for more concurrent connections

net.ipv4.ip_local_port_range = 1024 65000

# OS tunning

vm.swappiness = 0

# Increase system file descriptor limit

fs.file-max = 65535
```

Stats

- For small files, network BW is not a problem
- However network latency is a killer :(

Stats

Creation of **10000** empty files:

Local SSD	7.030s
MooseFS	12.451s
NFS + Ceph RBD	16.947s
OrangeFS	40.574s
Gluster distributed	1m48.904s

Stats

MTU	Congestion	result
MooseFS 10G		
1500	Scalable	10.411s
9000	Scalable	10.475s
9000	BBR	10.574s
1500	BBR	10.710s
GlusterFS 10G		
1500	BBR	48.143s
1500	Scalable	48.292s
9000	BBR	48.747s
9000	Scalable	48.865s

Stats

MTU	Congestion	result
MooseFS IPoIB		
1500	BBR	9.484s
1500	Scalable	9.675s
GlusterFS IPoIB		
9000	BBR	40.598s
1500	BBR	40.784s
1500	Scalable	41.448s
9000	Scalable	41.803s
GlusterFS RDMA		
1500	Scalable	31.731s
1500	BBR	31.768s

Stats

Creation of **10000** random size files:

MTU	Congestion	result
MooseFS 10G		
1500	Scalable	3m46.501s
1500	BBR	3m47.066s
9000	Scalable	3m48.129s
9000	BBR	3m58.068s
GlusterFS 10G		
1500	BBR	7m56.144s
1500	Scalable	7m57.663s
9000	BBR	7m56.607s
9000	Scalable	7m53.828s

Stats

Creation of **10000** random size files:

MTU	Congestion	result
MooseFS IPoIB		
1500	BBR	3m48.254s
1500	Scalable	3m49.467s
GlusterFS RDMA		
1500	Scalable	8m52.168s

Thank you!

Marian *HackMan* Marinov
mm@siteground.com

Thank you!
Questions?

Marian *HackMan* Marinov
mm@siteground.com

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

EUROPE

THE LINUX FOUNDATION