MySQL and ZFS

Yves Trudeau
Percona
Who am I?

• Principal architect at Percona since 2009 (10 years already…)
• With Sun Microsystems and MySQL before Percona
• Physicist by training
• I like to understand how things work
Why a talk on MySQL and ZFS?

- I like both and I couldn’t decide…
- They go along well
- They share many points in common
Plan

• A quick tour of ZFS
• Configuration guidelines for MySQL/ZFS
• A real world example
A tour of ZFS
ZFS Highlights

- Developed by Sun for Solaris
- Now in many platforms
- B-tree file storage, not just the directories
- 128 bits pointers!!!
- Files are split in records (b-tree leaves)
- Records can be compressed
- Copy-On-Write
- Native encryption
- Checksums and self-healing
ZPOOL

- Base unit of storage
- Made of block devices or even just files
- Disks, files, LV, mirror of disks, stripping, raidz, raidz2, raidz3…
- Filesystems from zpool
- A server → many zpools
- SLOG: Separated log device
- Cache devices, L2ARC
ZFS Filesystems

- A filesystem is:
  1. a profile of settings
  2. a mount point
  3. a snapshotable entity
- Settings adapted → expected workload
- Can be nested
- Can be based on a snapshot (clone)
ZVols

- A block device from ZFS
- Uber cool for virtual images
- Steps for a 3 nodes cluster:
  1. Create a base image on a Zvol
  2. Snapshot the ZVol
  3. Clone snapshot 3 times (yields 3 new ZVols)
  4. Start 3 VMs using the new Zvols

```xml
<disk type='block' device='disk'>
  <driver name='qemu' type='raw' cache='none' io='native'/>
  <source dev='/dev/zvol/data/vms/kvm_PXC2'/>
</disk>
```
The COW Magic

- ZFS never overwrites directly
- How ZFS overwrites a record?
  1. Writes it somewhere else
  2. De-references the old record → new record
  3. GC frees up the old record
- Easy snapshot (think InnoDB MVCC)
- Easy cloning
- Wonderful for backups
- Transactional!
ARC for Adaptive Replacement Cache

- Sophisticated file cache
- Configurable
- Can store compressed data
- Can be layered to disk (SSD/Flash) → L2ARC
Kernel Modules

- Many configuration parameters (ls /sys/modules/zfs/parameters/)
- Version 0.7.5 has 169…
- Examples:
  - zfs_arc_max: max size the ARC can be
  - zfs_arc_meta_limit: Caps the metadata limit in ARC
  - zfs_free_max_blocks: How fast the GC is going (InnoDB purge batch)
  - l2arc_write_max: how fast you allow writes to L2ARC
  - zfs_txg_timeout:max time span of a trx (think async writes)
Configuration Guidelines for MySQL/ZFS
When Should You Use MySQL/ZFS?

- For large compressible datasets
- Challenges with backup (mix of engines)
- Spare CPU capacity (compression)
- Not IO bound
- Active dataset fits L2ARC (compressed)
- To save your flash devices...
ZFS Configuration

• 2 file systems for easy snapshots
  ➔ /var/lib/mysql → The parent, configured for sequential ops
    ✔ recordsize = 128KB
    ✔ compression can be more aggressive (gzip)
  ➔ /var/lib/mysql/data → The dataset
    ✔ recordsize = InnoDB page size (likely 16KB)
    ✔ fast compressor like lz4
• Cache device (L2ARC) are great
• SLOG devices help with high durability requirements
MySQL Configuration

- `innodb_doublewrite = 0`
- `O_Direct?`
- InnoDB buffer pool? leave some Ram for the ARC
  - no L2ARC → target ARC 0.5% of the data set
  - 1TB of data ~ 5GB ARC
  - Not a hard rule
- `Datadir = /var/lib/mysql/data`
- `innodb_log_group_home_dir, log-bin, slow-log, relay-log to /var/lib/mysql`
Real World Examples
A DR MySQL Replica in Google Cloud

Dataset 700GB (2.5x compressible), fair replication traffic, all dataset is active (random primary keys)

<table>
<thead>
<tr>
<th>XFS</th>
<th>ZFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>n1-standard-2 (~68$/month)</td>
<td>n1-standard-2 (~68$/month)</td>
</tr>
<tr>
<td>1TB SSD (~175$/month)</td>
<td>local 375GB Nvme (30$/month)</td>
</tr>
<tr>
<td>Total: 243$/month</td>
<td>500GB standard disk (20$/month)</td>
</tr>
<tr>
<td></td>
<td>Total: 118$/month</td>
</tr>
</tbody>
</table>

ZFS saves 125$/month
A PXC Cluster in AWS

Dataset 2TB (2.5x compressible), needs more than 20k iops

XFS/i3
- 3x i3.4xlarge: $2700/month

XFS/EBS/io1
- 3x r5.2xlarge: $1080/month
- 3x 3TB 20k piops: $3900/month

ZFS/i3
- 3x i3.2xlarge: $1350/month
- 2TB SC1: $50/month

ZFS saves 1300$/month
## Will ZFS Really Perform Well?

Sysbench TPC-C workload emulation, GCE n1-standard-2 with local 375GB, Scale 300, 2 threads

<table>
<thead>
<tr>
<th></th>
<th>XFS</th>
<th>ZFS/Lz4</th>
<th>ZFS/Gzip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trx/s</td>
<td>110</td>
<td>69</td>
<td>59</td>
</tr>
<tr>
<td>Qps</td>
<td>3100</td>
<td>1954</td>
<td>1551</td>
</tr>
<tr>
<td>GB on disk</td>
<td>284</td>
<td>102</td>
<td>85</td>
</tr>
<tr>
<td>Used (%)</td>
<td>76%</td>
<td>39%</td>
<td>26%</td>
</tr>
</tbody>
</table>
Will ZFS Really Perform Well With L2ARC?

Sysbench TPC-C workload emulation, GCE n1-standard-2 with 500GB normal disk, 375GB local disk, Scale 300, 2 threads

<table>
<thead>
<tr>
<th></th>
<th>XFS</th>
<th>ZFS/Lz4/L2ARC</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRX/s</td>
<td>3</td>
<td>29 (l2arc warm)</td>
</tr>
<tr>
<td>QPS</td>
<td>87</td>
<td>830</td>
</tr>
<tr>
<td>Disk Usage</td>
<td>284 GB on disk 70% used</td>
<td>102 GB on disk 21% used</td>
</tr>
</tbody>
</table>

"● " marks the key performance indicators for each file system.
Conclusion

- MySQL and ZFS are great together
- Try, it is pretty easy
- Careful, you’ll get addicted
Thank You to Our Sponsors
Rate My Session

Introducing gh-ost: triggerless, painless, trusted online schema migrations

11:20 → 12:10
Matterhorn 2

Rate & Review

Tap a star to rate

Feedback (optional)

SUBMIT