SoS: Scale Out Samba: The Comeback of Ceph?

Ralph Böhme, Samba Team, SerNet

2024-04-17

Scaling Samba

Ceph as a replacement for CTDB key-value storage

Ceph offers a highly scalable and fault-tolerant storage system. Samba is already capable of sharing data located on the Ceph Filesystem, however scale-out sharing (the same data exposed by multiple Samba nodes) currently requires the use of CTDB for consistent and coherent state across Samba cluster nodes. In such a setup CTDB provides a clustered database with persistent key-value data storage and locking. Database usage is abstracted out via a generic dbwrap interface.

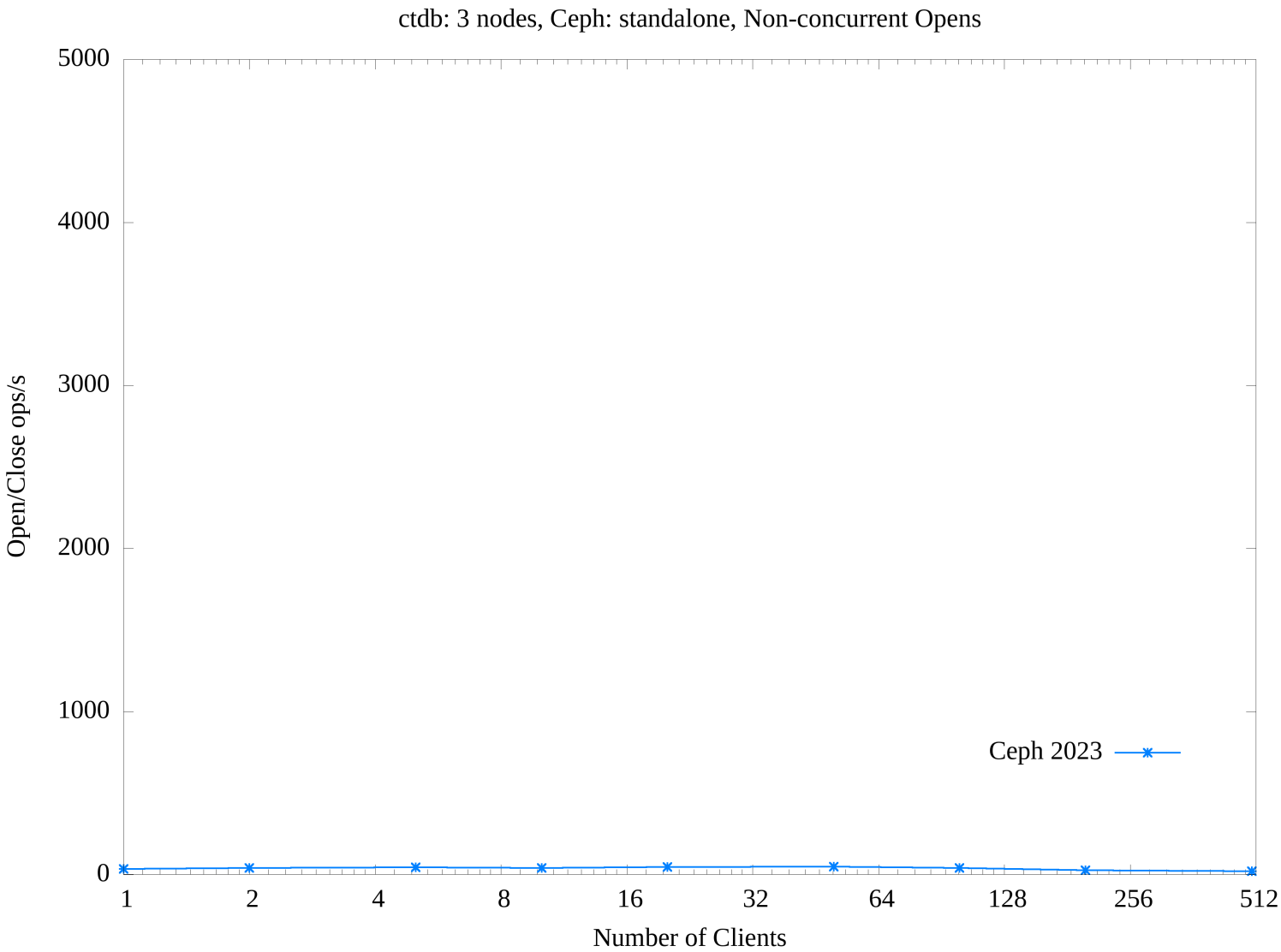
Ceph's librados library provides an API for the storage and retrieval of arbitrary key-value data via the omap functions. A watch/notify protocol is also provided as a mechanism for synchronising client state (locking). Key-value data stored in the RADOS back-end inherits the same redundancy features as regular objects, making it a potentially good candidate as a replacement for CTDB in scale-out Samba clusters.

– David Disseldorp, SUSE Hack Week 18, 2019

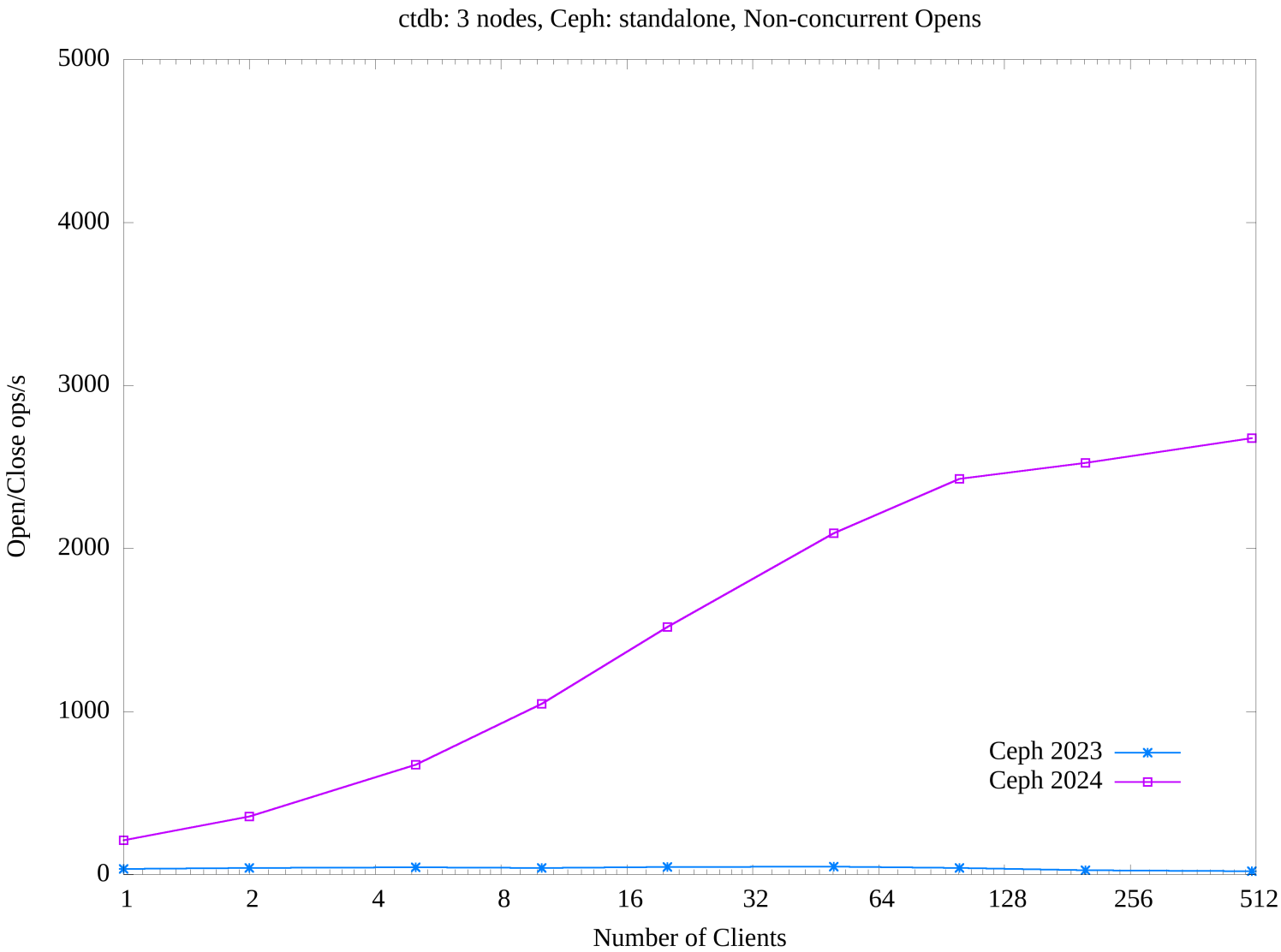
Client: open/close in a loop

```
$ smbtorture //172.18.111.10/test -U slow%x \
  smb2.bench.path-contention-shared \
  --unclist unclist-test.txt \
  --option=torture:timelimit=10 \
  --option=torture:nprocs=[1-500]
```

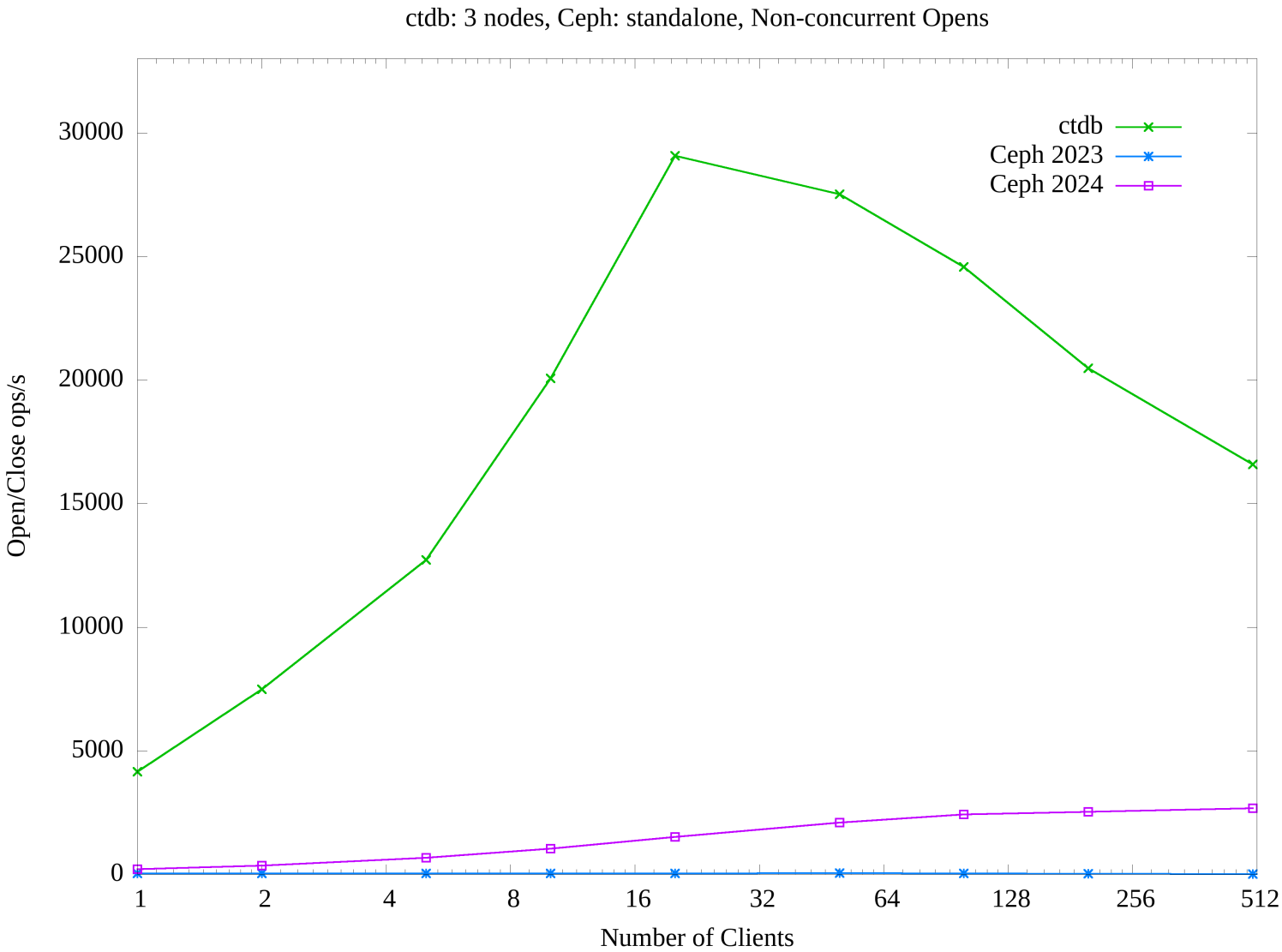
Performance: initial evaluation of Ceph as K/V in 2023



Performance: can Ceph do better?



Performance: Ceph vs ctdb



Why replace ctdb?

ctdb features

- fast, fast, fast
- but it cheats: data is not replicated

ctdb limitations

- ctdb has consistency, scalability and elasticity limitations
  - data is not replicated, required for SMB3 Persistent Handles
  - usecase is high-performance NAS in a single DC
  - not suitable for cloud SMB at scale
- Real world scalability: production max 16 nodes, 50k clients
- Elasticity: changing node count causes an expansive database redistribution
- Availability: support for mulit-datacenter and availabilty zones

Fix ctdb? Are there alternatives?

- There are many scalable Open Source distributed databases out there
- Can any of those fit the bill?
- Now returning to Ceph...

What changed?

2023

- using Ceph dbwrap backend from Samuel Cabrero
- Written in C, uses librados C library
- uses Ceph omap on a single object per Samba database

2024

- Ceph OSDs backed by SSDs, not spinning rust
- using Python Ceph rados backend for dbwrap_py
- allows tuning of object omap ratio

Ceph OMAP

What is a Ceph OMAP?

Omap is a key-value store, associated with an object, in a way similar to how Extended Attributes associate with a POSIX file. An object's omap is not physically located in the object's storage, but its precise implementation is invisible and immaterial to RADOS Gateway. In Hammer, one LevelDB is used to store omap in each OSD.

What is a Ceph OSD?

ceph-osd is the object storage daemon for the Ceph distributed file system. It manages data on local storage with redundancy and provides access to that data over the network.

Ceph OMAP vs Objects

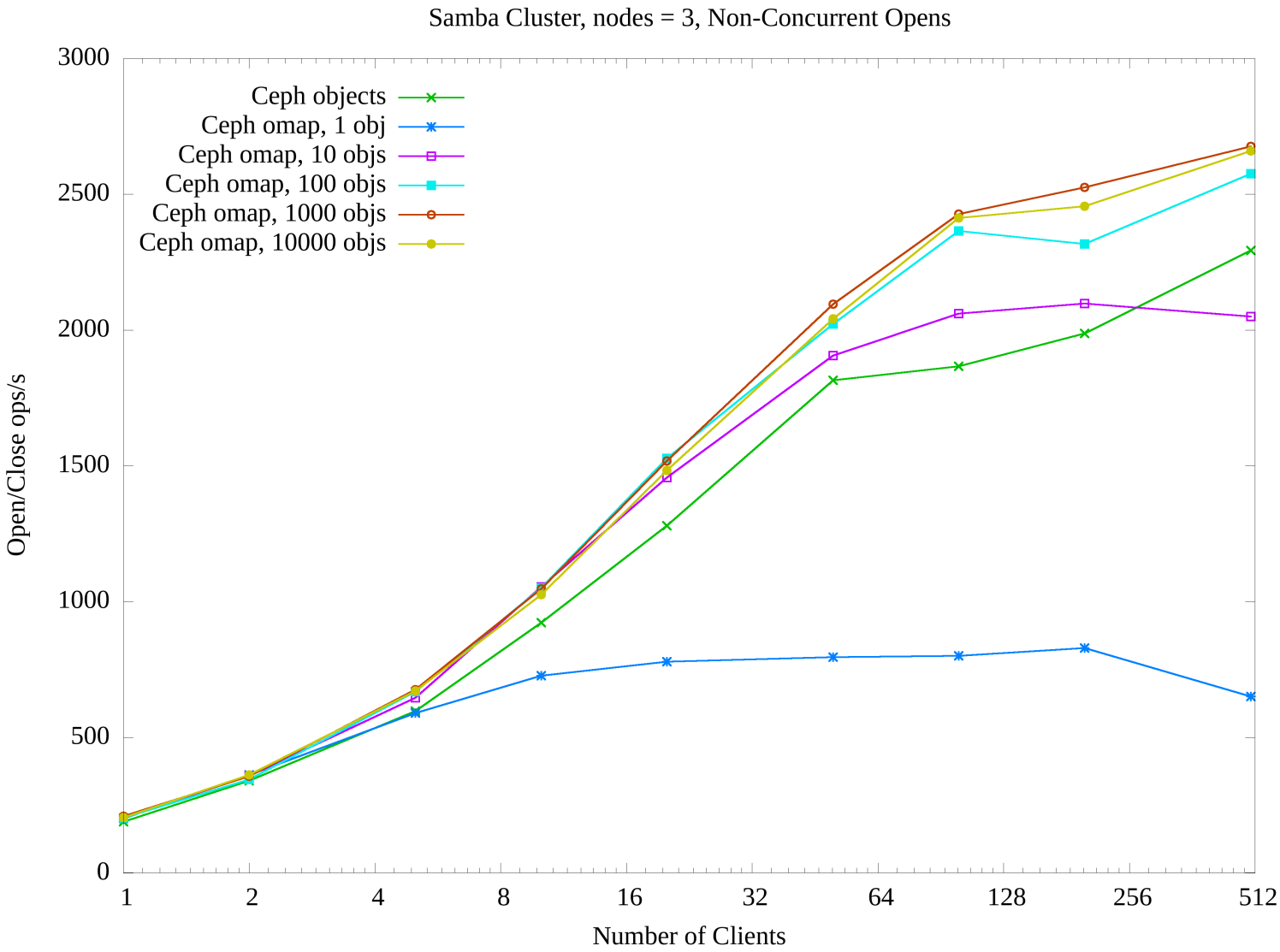
Choices. . .

1. Store values as objects
2. Distribute values across some amount of objects OMAPs

Downsides of using objects directly

- objects could use slower devices (e.g. HDDs)
- enumeration is slower
- metadata overhead for managing objects

Performance: Ceph objects vs OMAPs



Distributed Databases: ctdb? Ceph? Or what?

Requirements

Consistency

- Samba needs a database with strong consistency
  - for a non-transactional key/value store this means linearizability
  - for a transactional databases this means strict serializability

Atomic updates

- for atomic updates we need locks or single-key transactions

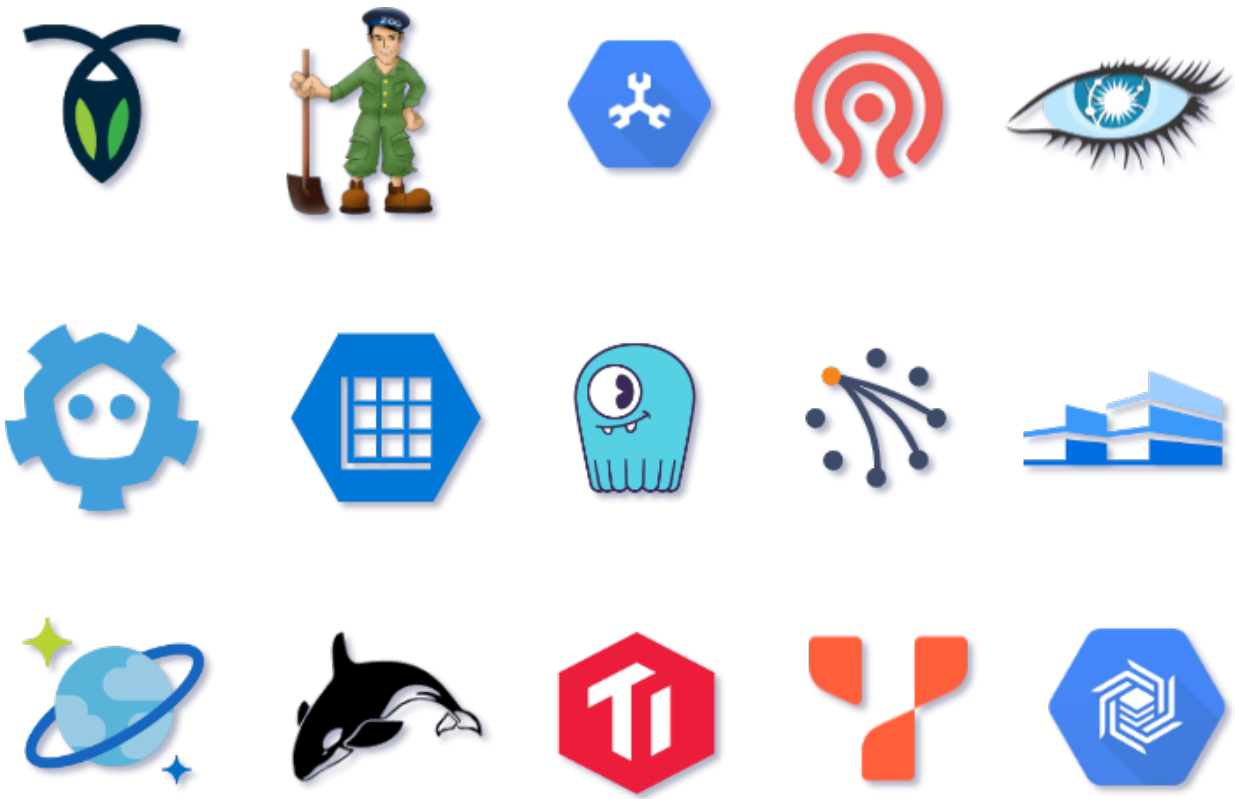
Distributed Locking

- locking is needed to serialize and isolate access to two resources:
  filesystem and database with file-handle state
- To implement locking we need either:
  native locking support from the DB, transactions or atomic compare-and-set

Python Bindings

- For fast prototyping

Zoo of Distributed Databases



CockroachDB, Zookeeper, Google Spanner, Ceph, Cassandra
etcd, Azure Table, Scylla, Riak, FoundationDB

Azure CosmosDB, Apache Hbase, TiKV, Yugabyte, Google Bigtable

dbwrap_py

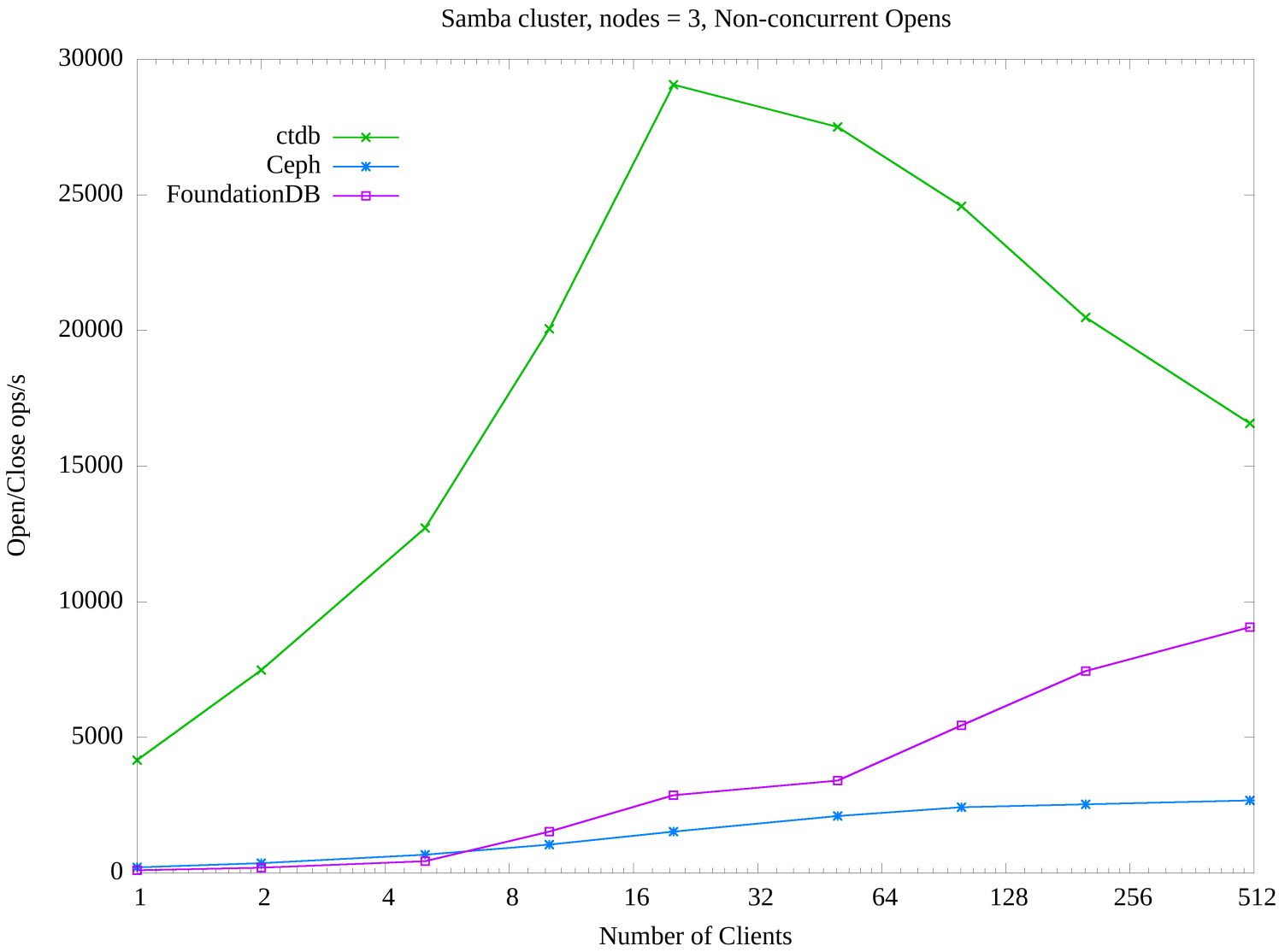
- Simplify database adapter development: use Python
- Just 1000 lines of C code (without txn support)
- Using Python for the backend allows rapid prototyping and testing

```
$ wc -l python/samba/samba3/dbwrap_py_*
  338 python/samba/samba3/dbwrap_py_cassandra.py
  170 python/samba/samba3/dbwrap_py_ceph.py
  414 python/samba/samba3/dbwrap_py_etcd3.py
  303 python/samba/samba3/dbwrap_py_fdb.py
   47 python/samba/samba3/dbwrap_py_tdb.py
```

FoundationDB

- Key-value store with transaction support
- Highly scalable, used by Apple (iCloud) and Snowflake (s3 object metadata)
- One of the very few open source distributed DBs with a C client

Performance: Ceph vs FoundationDB vs ctdb



Discussion

IOPS

fio rw=75%, 4k-random

- fdb: r=12k,w=4k IOPS per node
- Ceph: r=27k, w=9k IOPS per OSD, 3 OSDs
- Reported fio IOPS = iostat IOPS

smbtorture IOPS (nprocs=500):

- CPU: 30% idle (4 cores)
- ceph iostat: avg 25k IOPS
- But iostat reports underutilized disk:

```
$ iostat -xd 1 dm-0 sdb
Device   r/s     w/s  wrqm/s aqu-sz  %util
dm-0   32.00 8234.00    0.00   4.96 100.00
sdb    32.00 1995.00 5239.00   0.33 100.00
```

- It seems Ceph OSD IO scheduling does not saturate SSDs
- Reportedly partitioning SSDs and running an OSD per partition helps

Conclusions

And the winner is...

- Still FoundationDB over Ceph
- Both Ceph and FoundationDB performance severely limitted by test hardware (Ceph: VMs on a single physical server, fdb: cloud VMs)
- Both Ceph and FoundationDB performance depends heavily on disk IOPS
- With NVMe SSDs, servers can achieve more then 10m IOPs per server
  - in these tests we had a fraction of this (Ceph: 36k IOPS, fdb: 16k IOPS)
- Expect SMB performance to be at least a magnitude faster on fast hardware

Q&A & Links

Links

- dbwrap_py git branch
- How to maximize the OSD effective queue depth in Ceph?
- NVMe SSD partitioning

Thank you!
Questions?

Ralph Böhme
slow@samba.org
rb@sernet.de