File and Metadata Replication in XtreemFS

Björn Kolbeck
Zuse Institute Berlin

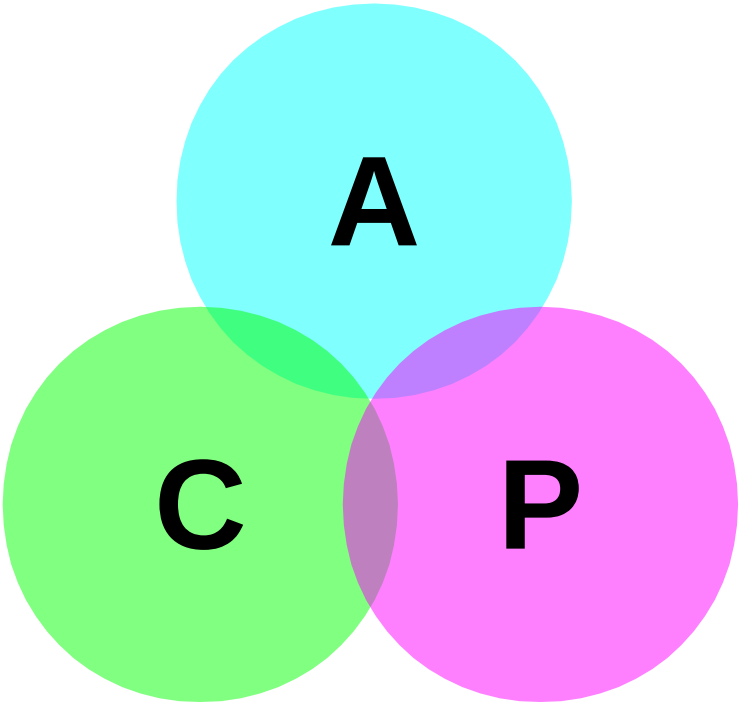
Why Replicate ?

- fault tolerance
  - mail server
  - source repository
- bandwidth
  - start 1,000 VMs in parallel
  - grid workflows
- latency
  - local repositories (climate data, telescope images)
  - HSM: fast (disk) vs. slow (tape) replicas

The CAP Theorem

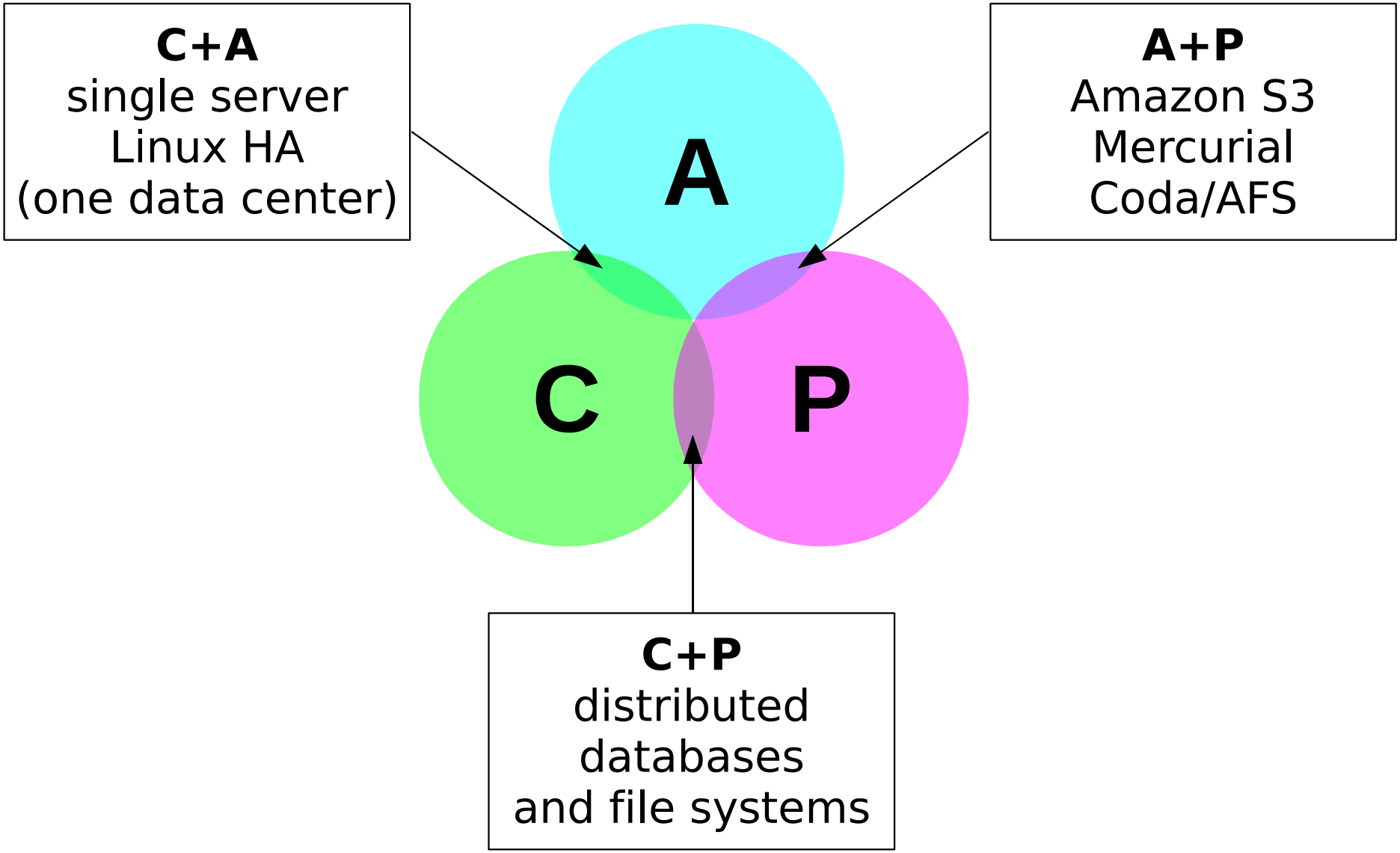
- **C**onsistency
- **A**vailability
- **P**artition tolerance

- "dernier cri"
  A+P (eventual consistency)

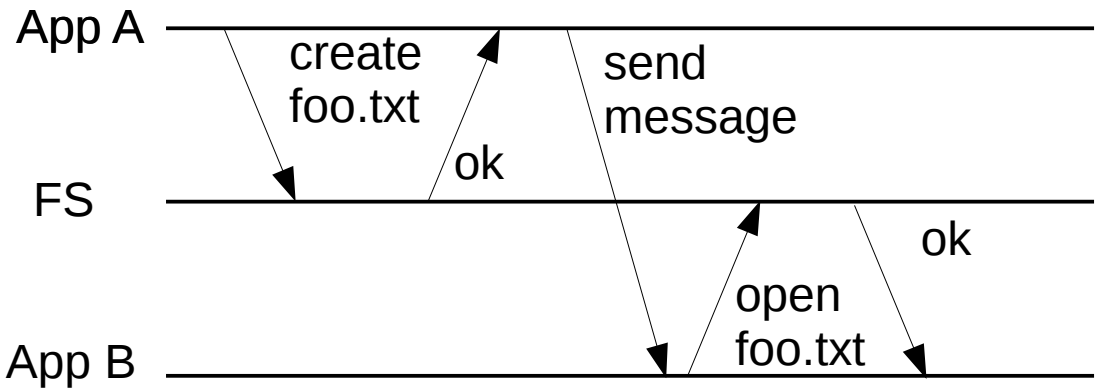


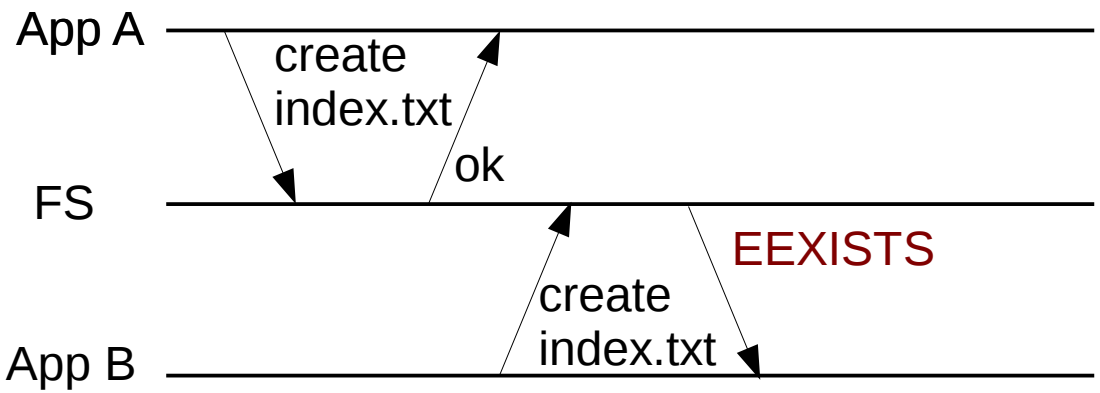
Brewer, Eric. Towards Robust Distributed Systems. PODC Keynote, 2004.

CAP: Examples



File System: Expected Semantics





File System: Consistency

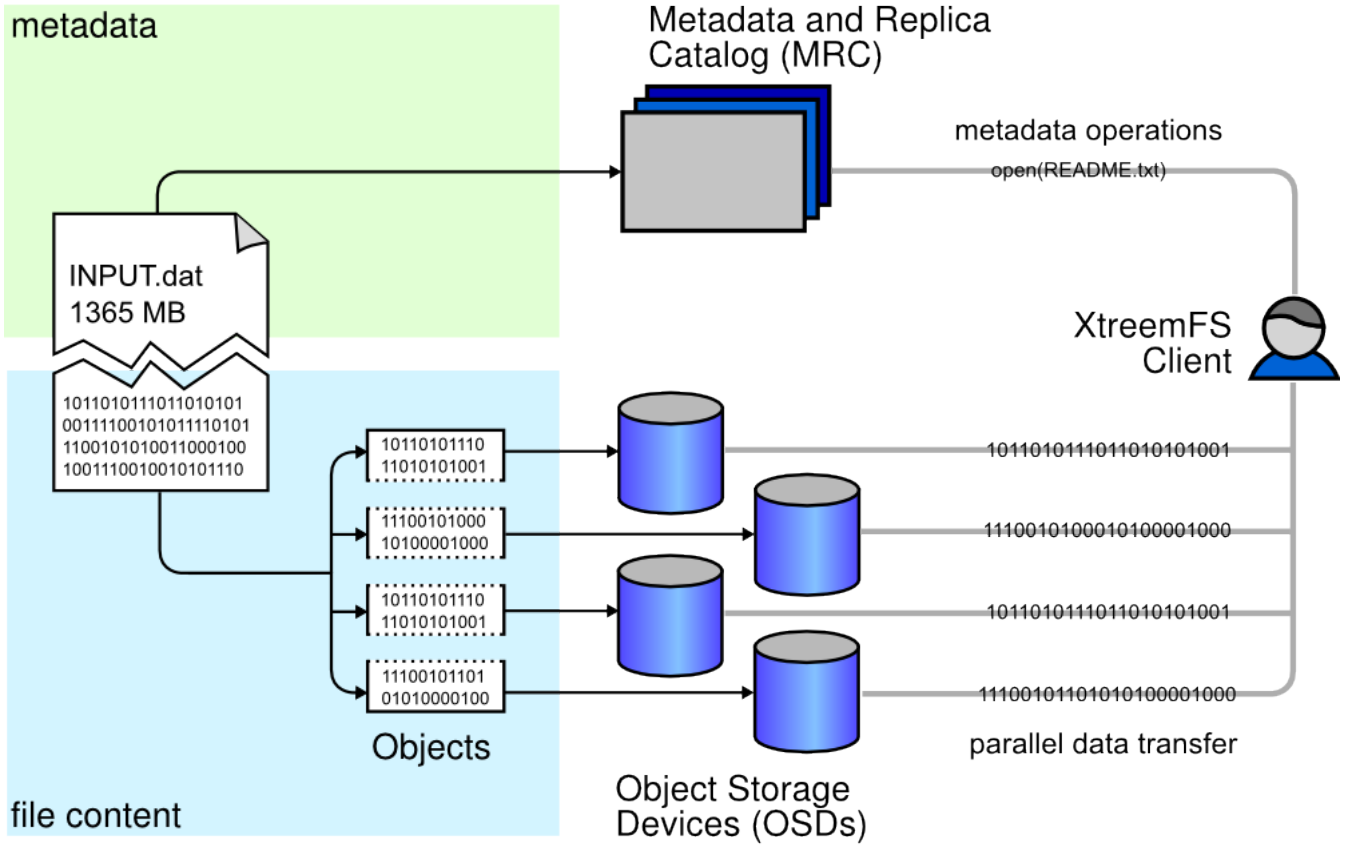
- linearizability (metadata and file data)
  - communication between applications / users
- atomic operations (metadata only)
  - unique file names (create, rename)
  - used by real-world applications
    e.g. dovecot
- expensive

File System: Do we really need consistency?

- A+P = conflicts
  - name clashes
  - multiple versions
- A+P vs. POSIX API
  - can't resolve name clashes
  - no support for multiple versions
  - no interface to resolve conflicts
- A+P vs. Expectations
  - developers assume consistency
  - synchronization

XtreemFS

- distributed file system
- object-based design
- "POSIX semantics"
- focus on replication (grid, cloud)



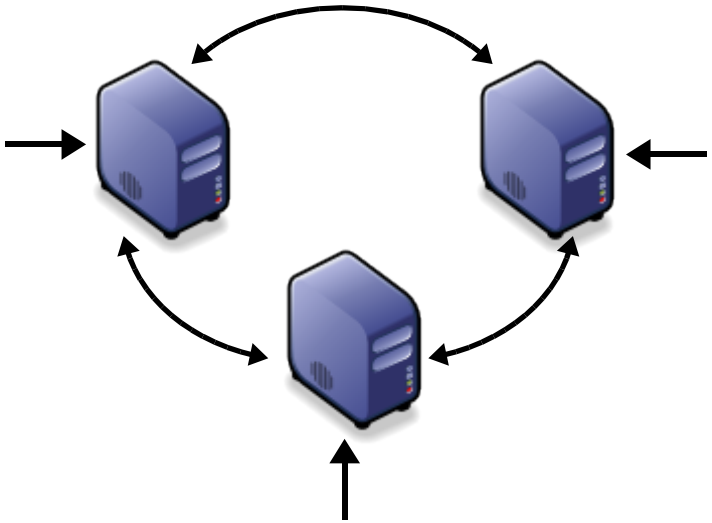
Two problems – one solution

1. Metadata replication
   - problem: **bottleneck**
   - replication algorithms
   - "relax" requirements
   - our solution
2. File data replication
   - problem: **scale**
   - our solution
   - central lock service
3. Other file systems

Metadata: How to replicate?

Replicated State Machine (C+P)

- Paxos



+

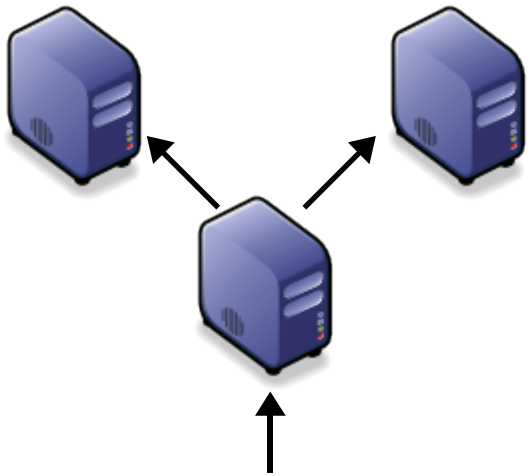
- no primary/master
- no SPOF
- no extra latency on failure

–

- slow
  two round trips
- needs distr. transaction
- difficult to implement

Metadata: How to replicate?

Primary/Backup (C+P)
- replicated databases



+

- fast
  write = 1RT, read = local
- no distr. transactions
- easy to implement

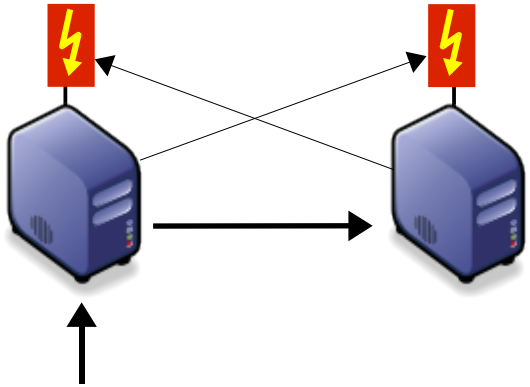
–

- primary failover
  short interruption
- primary = bottleneck

1. Metadata: How to replicate?

Linux HA (C+A)

- heartbeat signal + STONITH
  shared storage
- Lustre failover



+

- can be added "on-top"

–

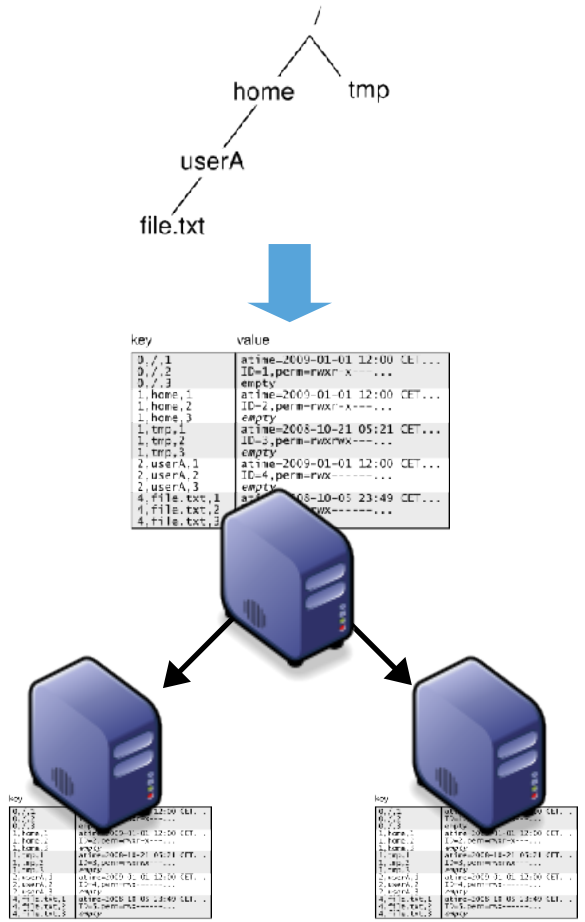
- still SPOFs: STONITH...
- only for clusters
- passive backups

1. Metadata: "relax"

- read all replicas = sequential consistency
  - **`stat`**, **`getattr`**, **`readdir`** (50 - 80% of all calls)
  - load balancing
  - upper bound on "staleness"
- write updates asynchronously
  - ack after local write
  - max. window of data loss
  - similar to sync in PostgreSQL

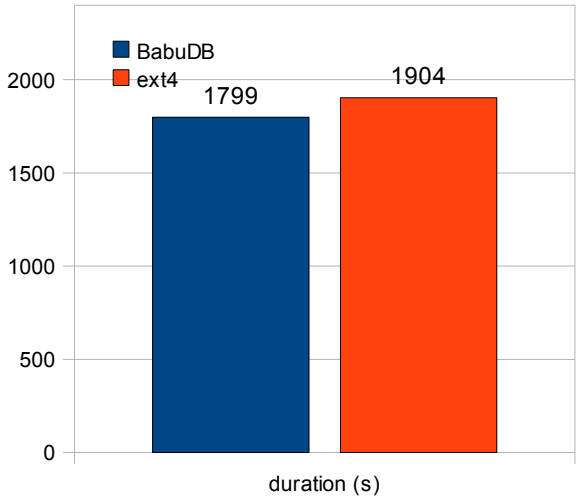
1. Metadata: Implementation in XtreemFS

- map metadata on a flat index
- replicate index with primary/backup
  - use leases to elect primary
  - replicate insert/update/delete
- future work:
  - weaker consistency for some ops
    e.g. chmod, file size updates
  - upper bound on "staleness"

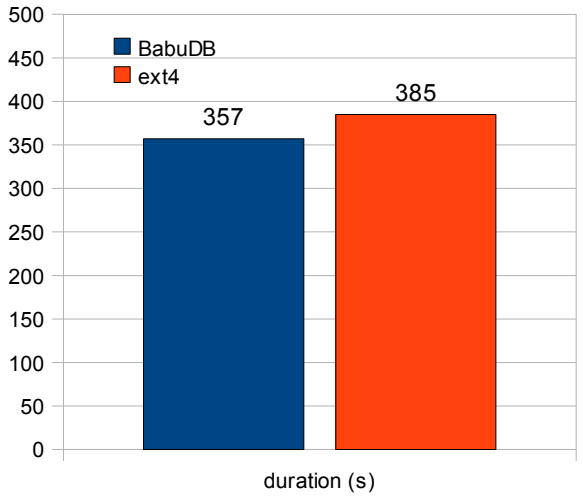


1. Metadata: Excursion — Flat index vs. Tree

- database backend (BabuDB, LSM-Tree based)
- ext4 (empty files)



linux kernel build



IMAP trace (docevot imapstress)

➔ competitive performance

2. File Data: Expected Semantics

- same as metadata
  - but no atomic operations
- many applications require less
  - read-only files / write-once
  - single process reading/writing
  - explicit fsync

2. File Data: Implementation in XtreemFS

- write-once: separate mechanism
  - more efficient
  - support for partial replicas
  - large number of replicas
- read-write: primary/backup
  - use leases for primary failover
  - requires service for lease coordination, e.g. a lock service

2. File Data: Problem of Scale

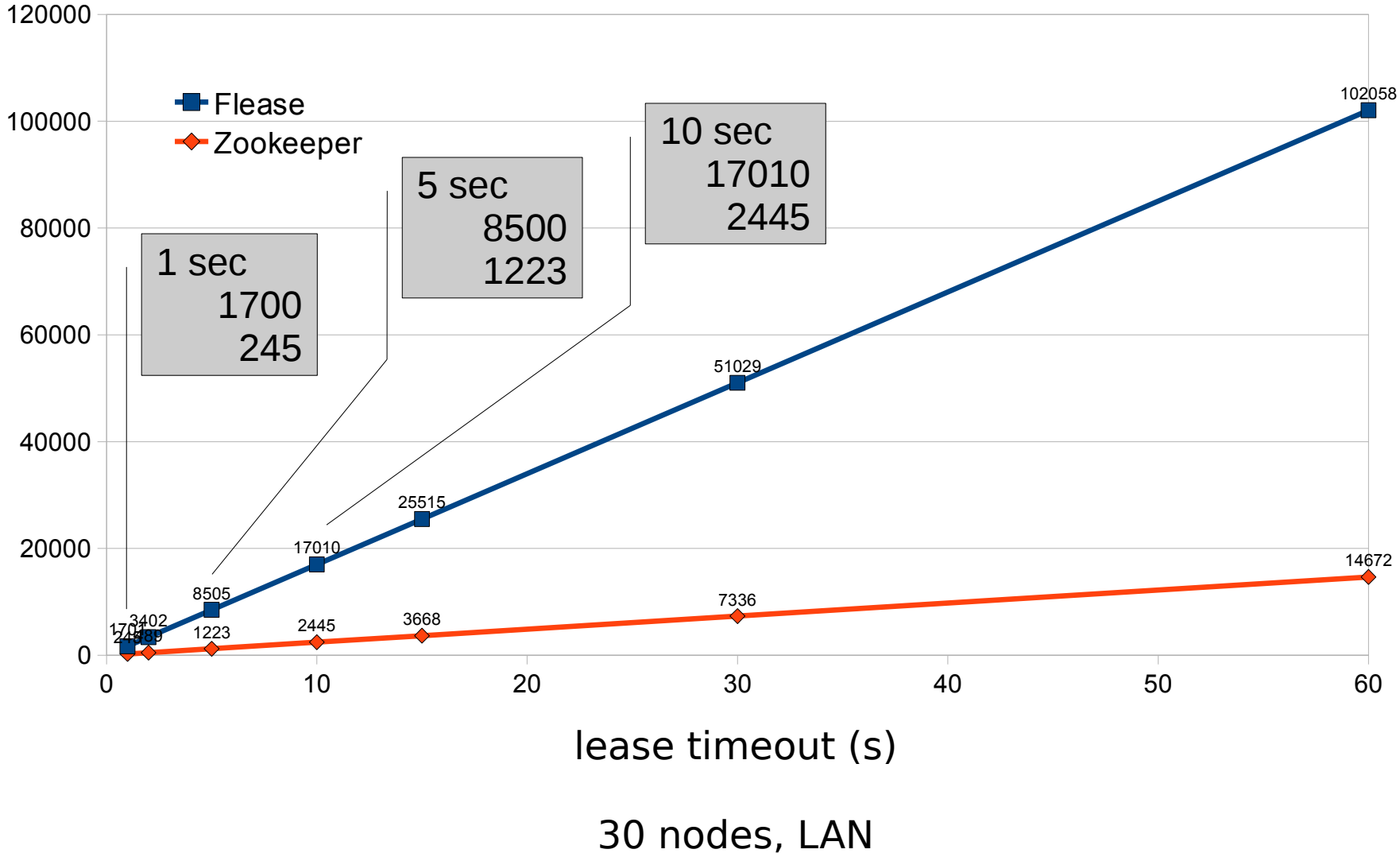
- Large number of storage servers
- Large number of files

- Primary per open file?
- Primary per partition?
- Long leases timeouts, e.g. 1min?

2. File Data: How to coordinate many leases?

- Flease: decentralized lease coordination
  - no central lock service
  - coordinated among storage servers holding a replica
- numbers:
  - Google's Chubby: ~640 ops/sec
  - Zookeeper: ~7,000 ops/sec
  - Flease: ~5,000 ops/sec (3 nodes), ~50,000 ops/sec (30 nodes)

2. File Data: Max. number of open files/server



Replication: Other File Systems

	Metadata	File data
Lustre	Linux HA	Linux HA
CEPH	-	primary/backup + central cfg. service and monitoring
GlusterFS	-	RAID 1
HDFS	-	write-once

Replication: Lessons Learned

- event-based design
  → no message re-ordering
- separation of replication layers
  → simplified implementation, testing
- no free lunch – consistency across data-centers is expensive

Thank You

- http://www.xtreemfs.org
- upcoming release 1.3 includes replication

- XtreemFS is developed within the XtreemOS project. XtreemOS is funded by the European Commission under contract #FP6-033576.