Durability with BookKeeper

Flavio Junqueira
Yahoo! Research, Barcelona

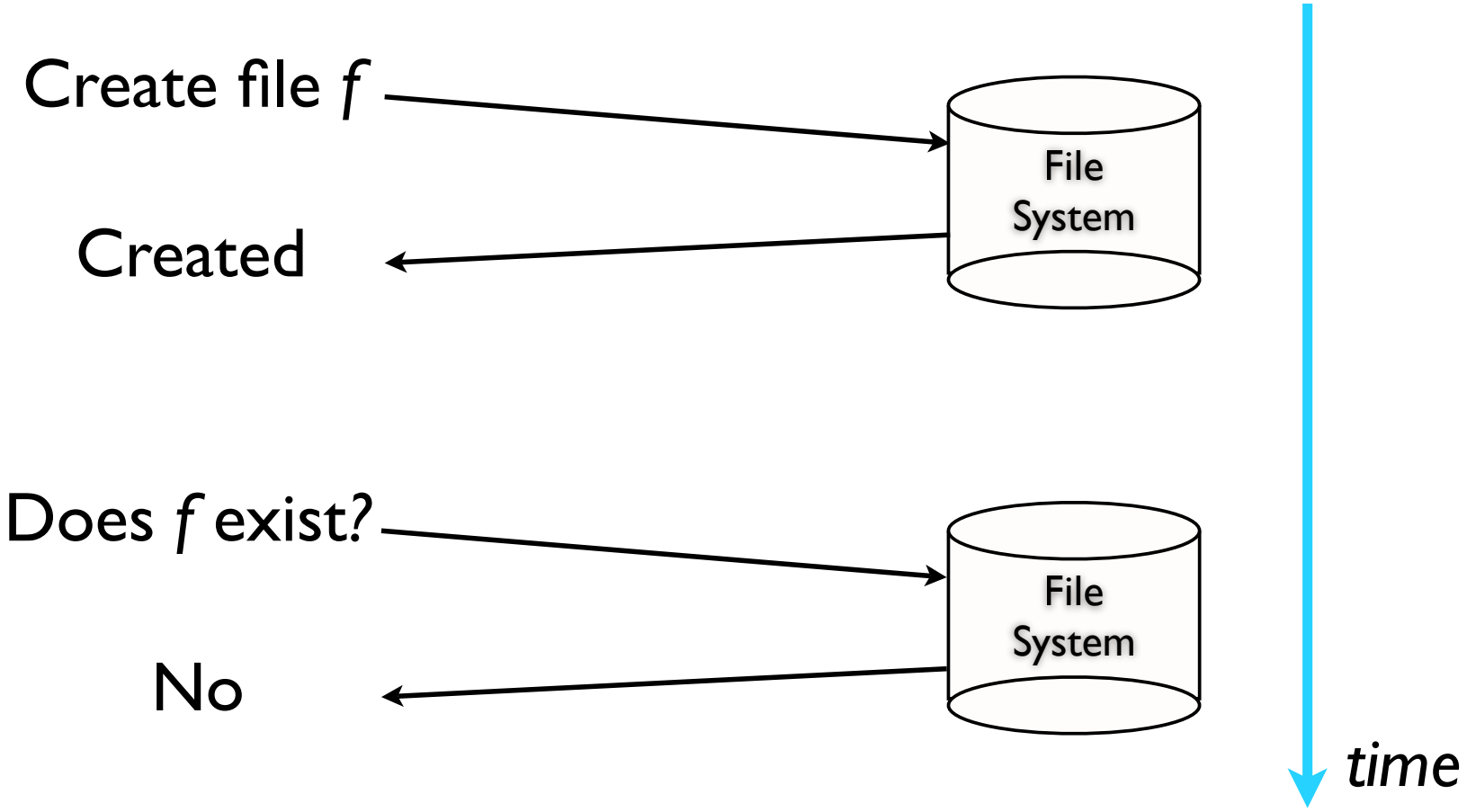
with Ben Reed and Ivan Kelly

LADIS 2012

Context

- ZooKeeper
  - ✓ Targets read-dominated workloads
  - ✓ Writes are efficient, but shouldn't be on the critical path of common operations
- Some applications need efficient writes
  - ✓ Durability

No Durability



No Durability

New status: scratching my nose

Posted

New status: scratching my elbow

Posted

My latest status?

scratching my nose

time

Recoverability

- Stateful systems
- Ability to recover after
  - ✓ Crashes
  - ✓ Planned outages
- Writes to stable store

Examples

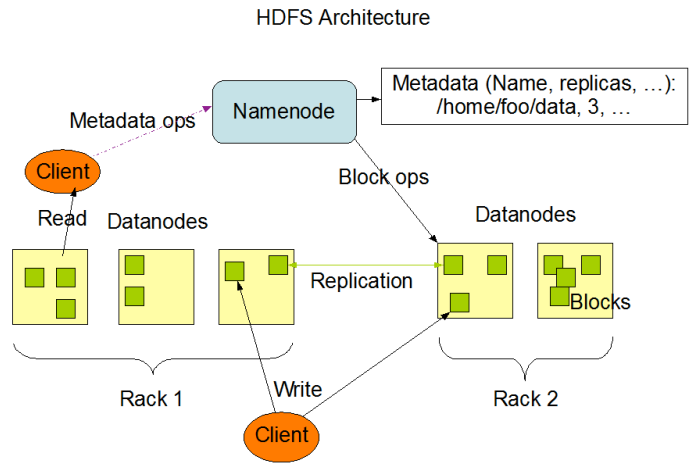
- HDFS namenode
- HBase region server
- ZooKeeper
- Pub-sub brokers
  - ✓ ActiveMQ brokers
  - ✓ Hedwig hubs

Examples

- HDFS namenode
- HBase region server
- ZooKeeper
- Pub-sub brokers
  - ✓ ActiveMQ brokers
  - ✓ Hedwig hubs

HDFS at a glance

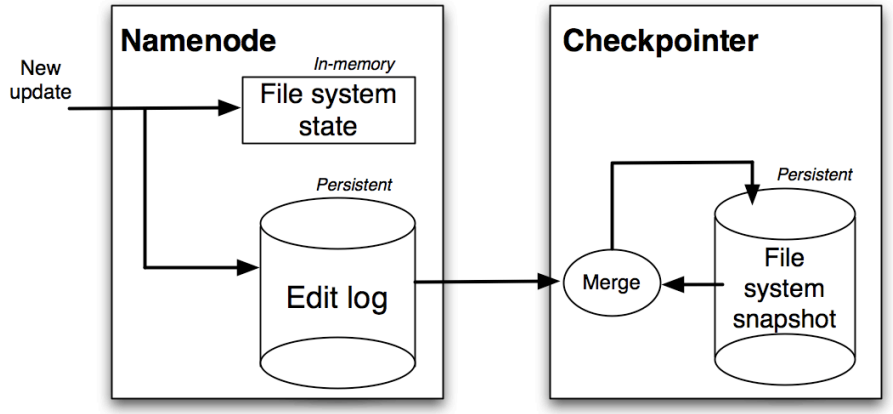
- Main components: namenode and datanode
  - ✓ Single name node (no considering federation)
  - ✓ A number of data nodes
- Namenode
  - ✓ Manages FS namespace
  - ✓ Regulates access to the FS
  - ✓ Mapping of blocks to data nodes
- Datanode
  - ✓ Stores blocks
  - ✓ Serves reads and writes



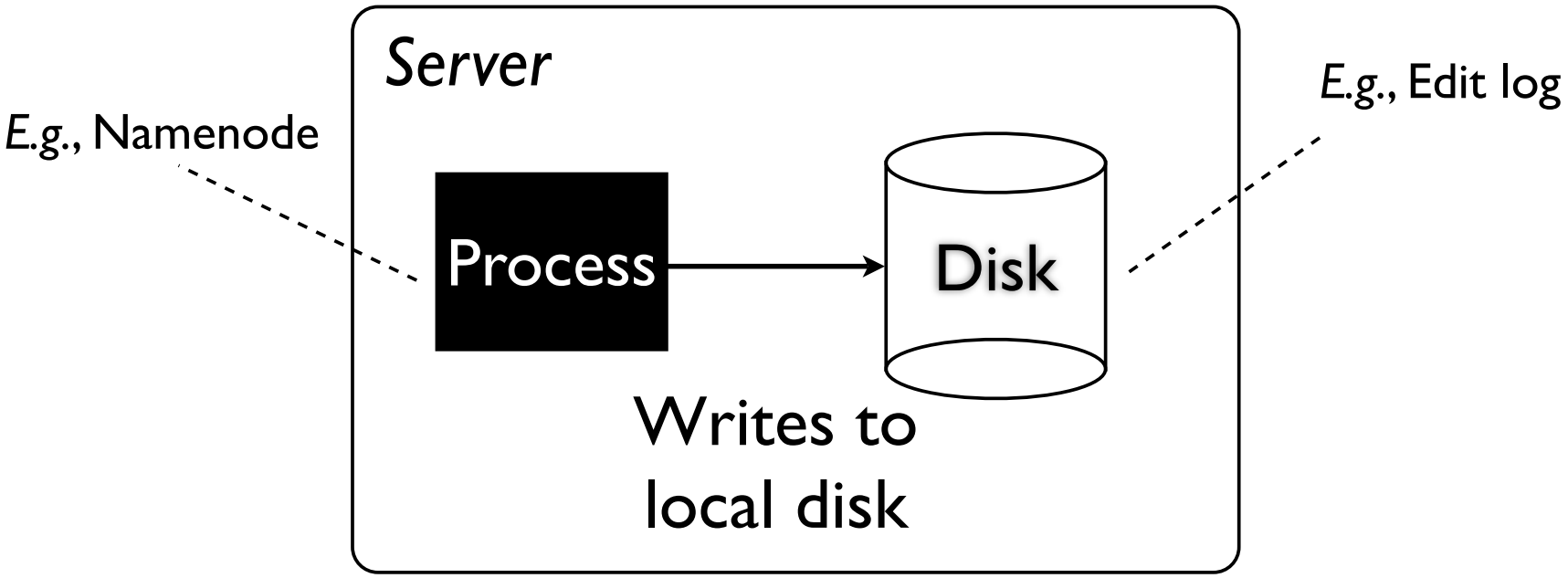
http://hadoop.apache.org/common/docs/current/hdfs_design.html

Namenode

- File system state
  - ✓ Metadata, block map
  - ✓ In memory
- Checkpoint
  - ✓ On disk
  - ✓ Snapshot of the service state
- Edit log
  - ✓ Persists changes to the file system metadata
  - ✓ Written to disk

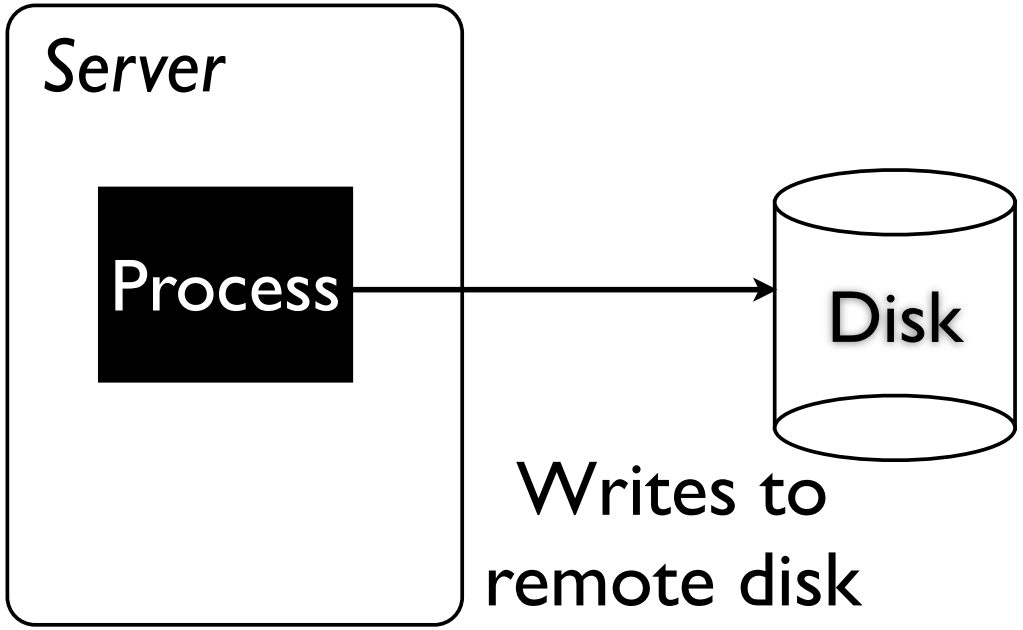


Recoverability



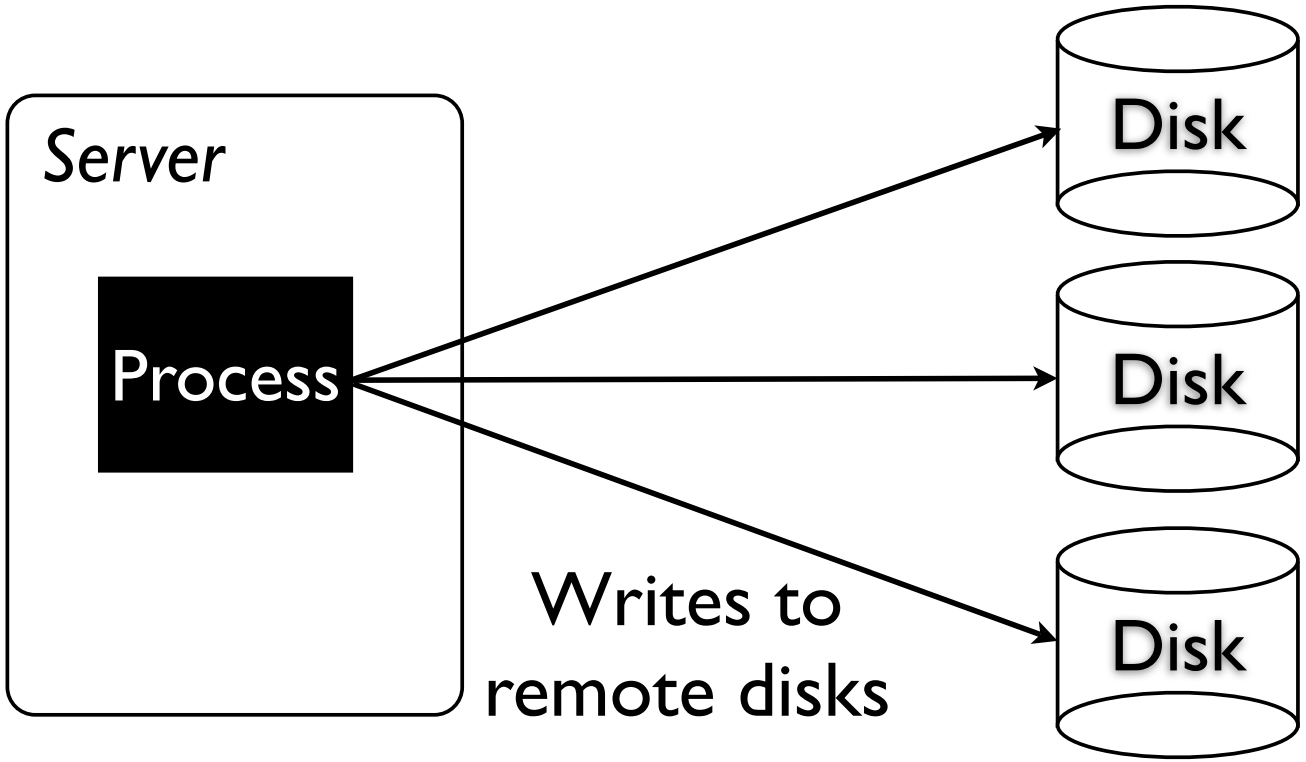
Not recoverable upon a crash

Recoverability



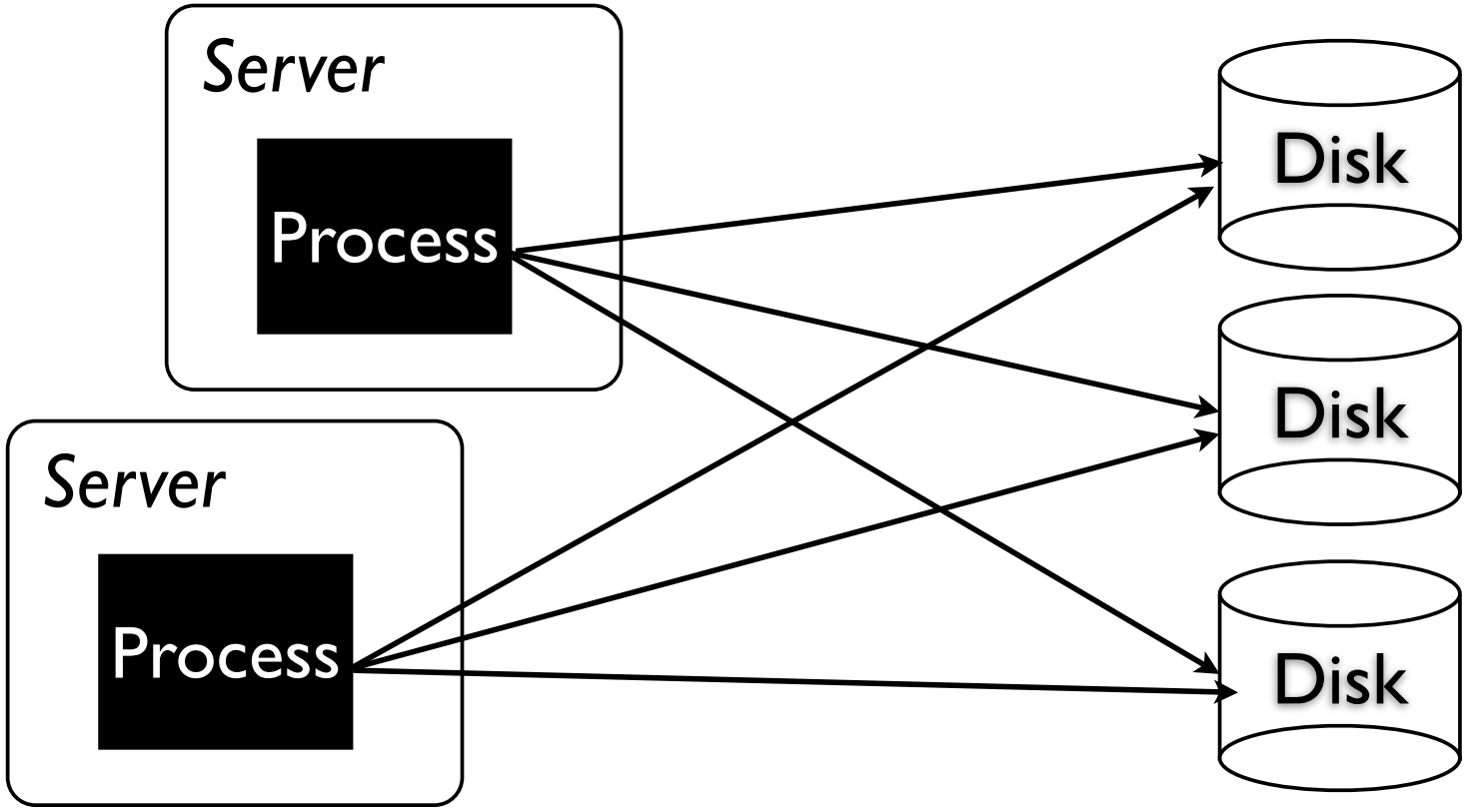
Better, but single copy of data

Recoverability



Too much infrastructure for a single application

Recoverability



Shared infrastructure

Remote writes

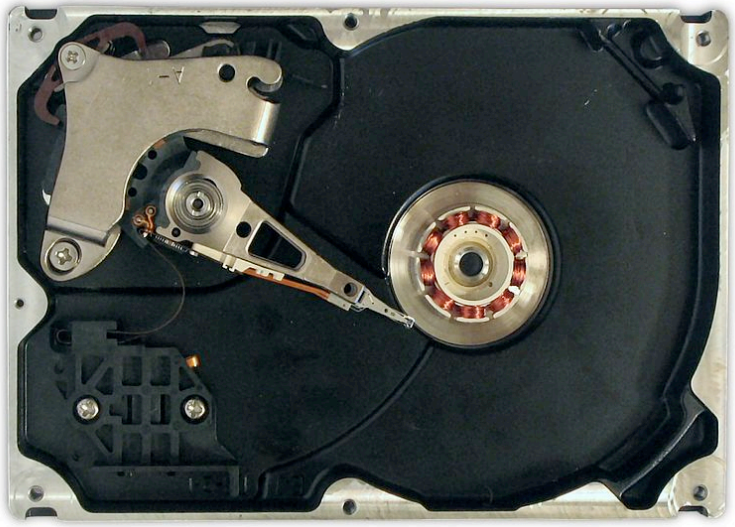
- Typically NFS mounts
  - ✓ *E.g.*, NetApp filers
  - ✓ Enterprise grade, robust devices
  - ✓ Not practical to have many
  - ✓ Single point of failure
- Commodity hardware is preferable

So far...

- Remote writes
  - ✓ For availability
- Writes to multiple disks
  - ✓ Tolerates disk crashes
- Commodity hardware
- Are we done?

Writes to disk

- Access time
  - ✓ seek time +
  - ✓ rotational latency +
  - ✓ transfer time
- Seek time typically dominates



Efficiently writing to disk

- Many tricks
  - ✓ Sequential writes
  - ✓ Group commits
  - ✓ Preallocation
  - ✓ Indexing for efficient reads
  - ✓ Independent disks for minimal performance impact

http://zookeeper.apache.org/bookkeeper

What's BookKeeper?

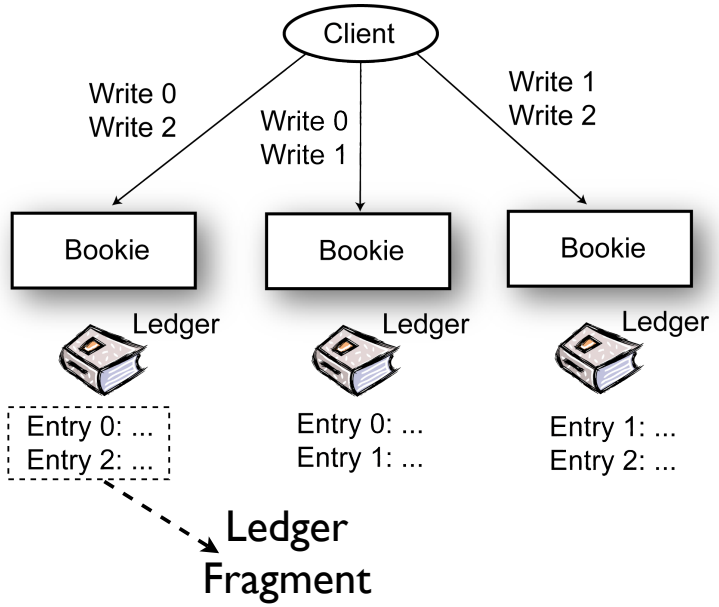
- Shared storage
- Sequences of byte arrays
- Data is replicated
- Writes are striped
- Many processes can access it

BookKeeper

- Shared storage for logs
  - ✓ Single Writer/Multiple Reader
- Design goals
  - ✓ Efficient sequential writes
  - ✓ Fault tolerance
  - ✓ Scalability

BookKeeeper architecture

- **Bookie**: Storage node
- **Ledger**: log file
- **Ensemble**: group of bookies storing a ledger
- Writes to quorums of Bookies
- Parallel writes to quorums
- Reads from the same quorum



API at a glance

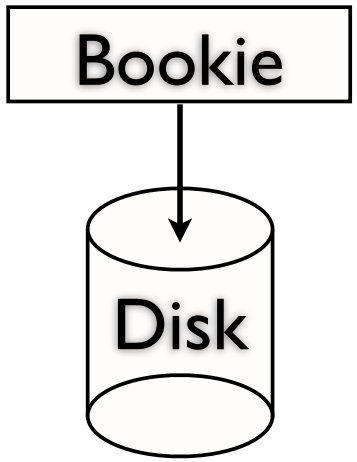
- `createLedger`
- `addEntry`
- `closeLedger`
- `openLedger`
- `readEntries`

Bookies

- BookKeeper storage nodes
- Do not communicate with other bookies
- Store ledger fragments efficiently

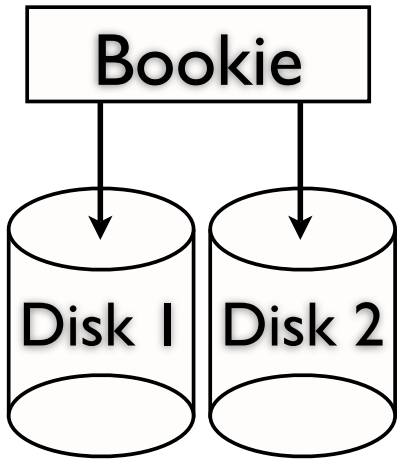
Bookie - Design 1

- Single disk
- Bookie writes sequentially to one disk
  - ✓ Appends ledger entries
- Great write performance
- Read performance
  - ✓ Interferes with writes



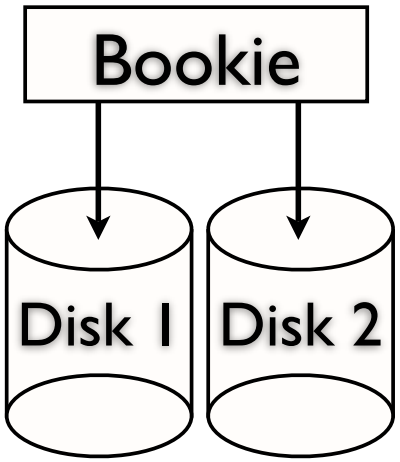
Bookie - Design 2

- Separates read and write traffic
- Disk 1 (Txn log device)
  - ✓ Sequential and synchronous writes
- Disk 2 (Ledger device)
  - ✓ One file per ledger
  - ✓ Asynchronous writes

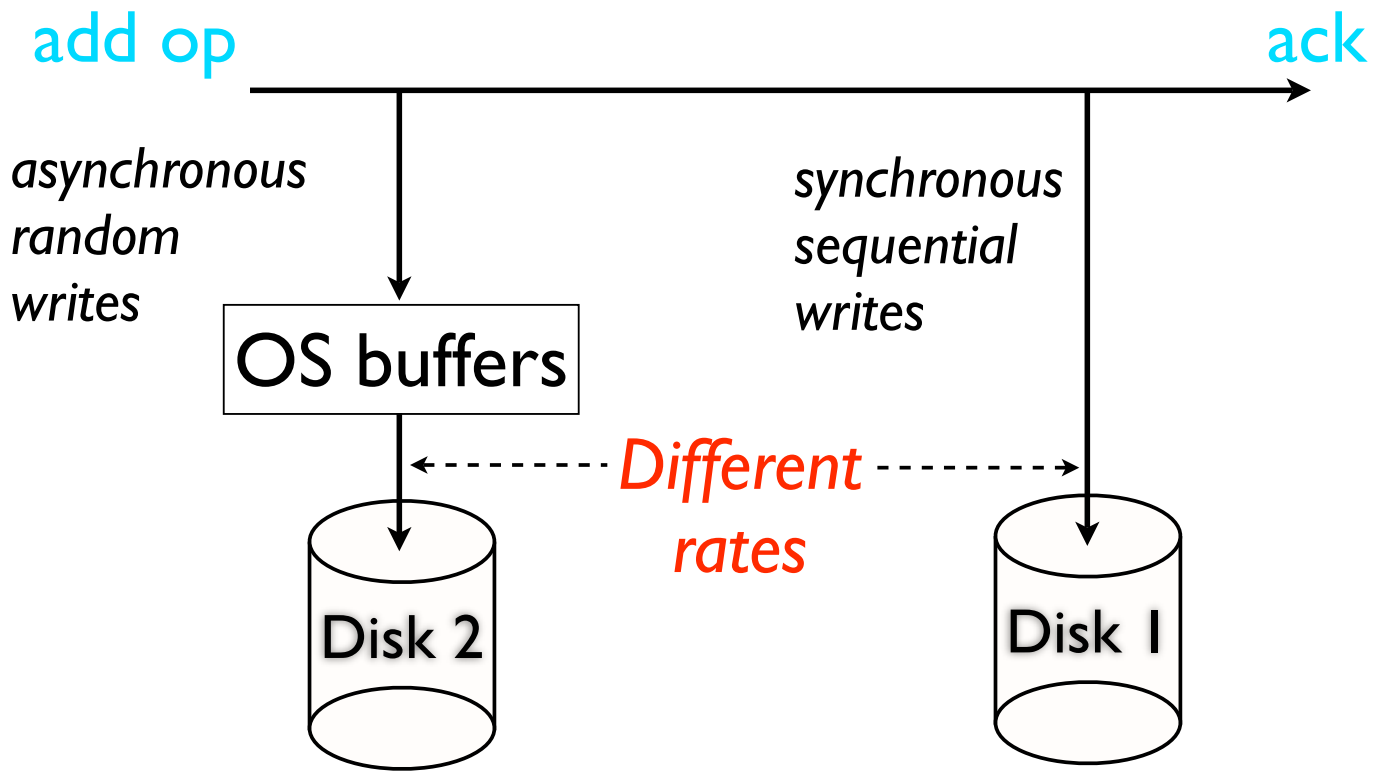


Bookie - Design 2

- Traffic
  - ✓ Concurrent ledgers
- Disk 2 (Ledger device)
  - ✓ Random seeks
  - ✓ Writes to disk 2 are slower
  - ✓ Eventually causes a convoy effect

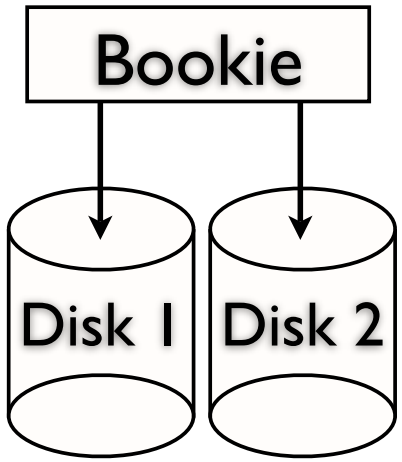


Bookie - Design 2



Bookie - Design 3

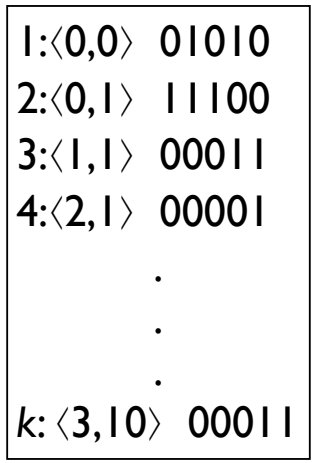
- Separates read and write traffic
- Disk 1 (Txn log device)
  - ✓ Sequential and synchronous writes
- Disk 2 (Ledger device)
  - ✓ Asynchronous writes
  - ✓ Sequential writes



Ledger device

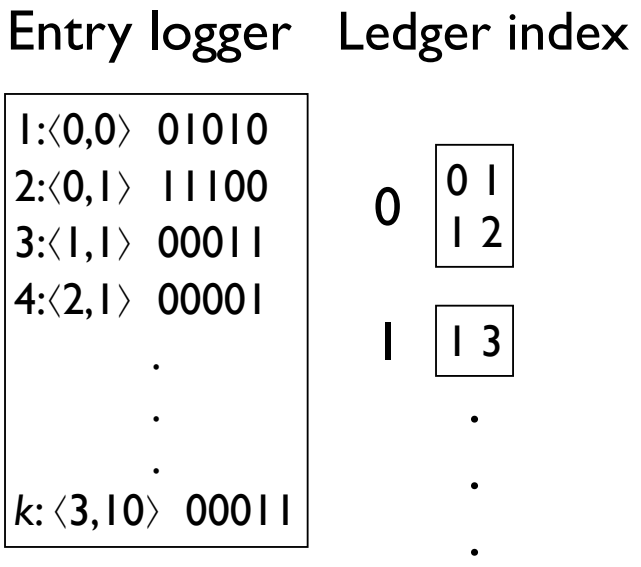
- Entry logger
  - ✓ Writes new entries sequentially
  - ✓ One for all active ledgers
- What about reads?
  - ✓ Scan the logger is not viable

Entry logger

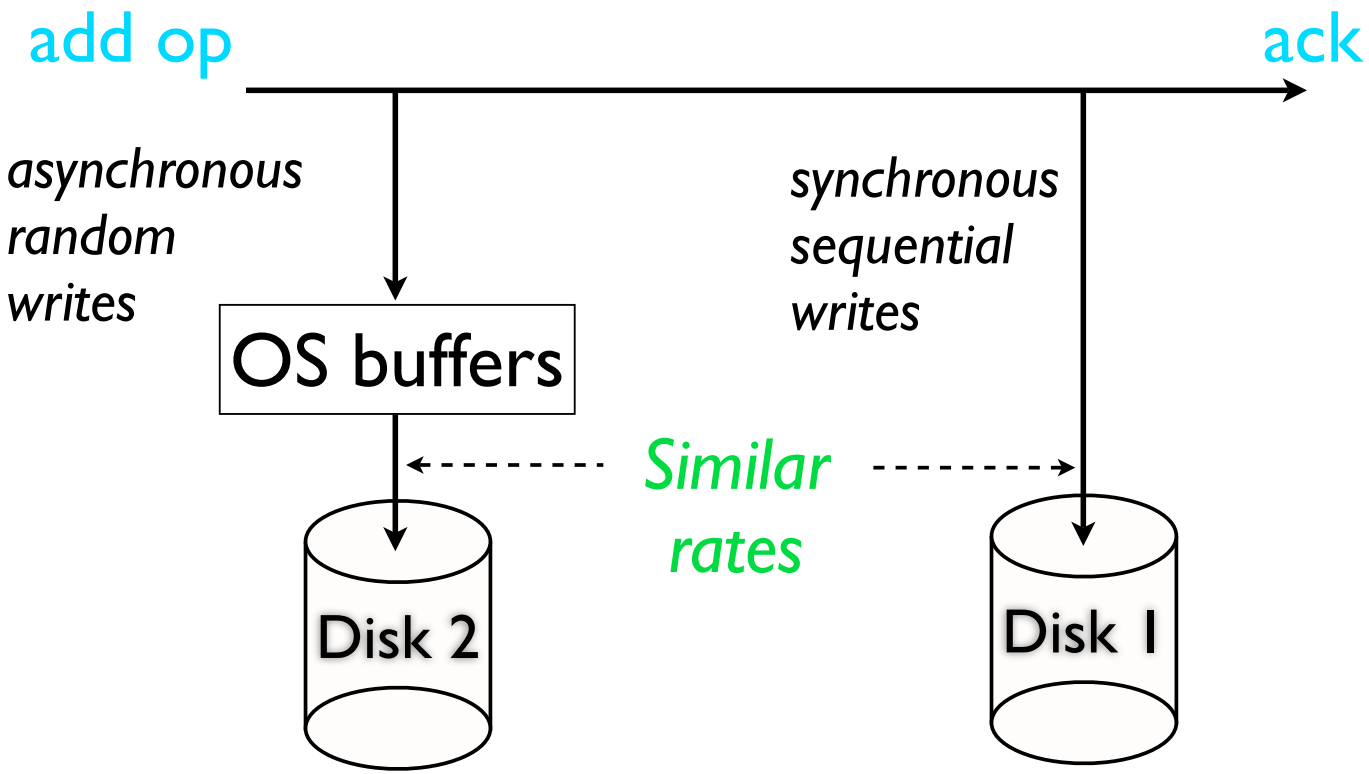


Ledger device

- Ledger index
  - ✓ entry id → log position
  - ✓ Might induce random seeks
- Ledger index cache
  - ✓ Caches index pages
  - ✓ Flushes pages to disk
  - ✓ Longer sequential writes (8k bytes)

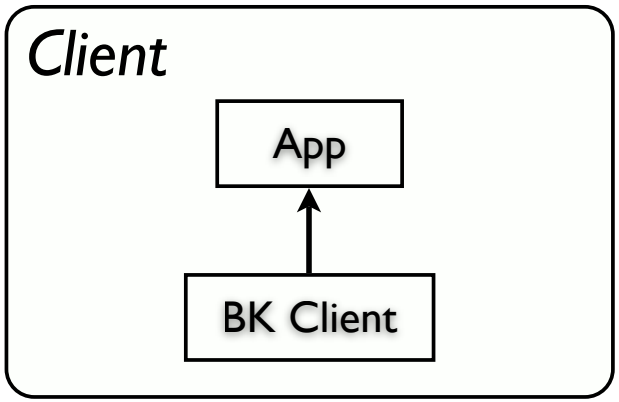


Bookie - Design 3



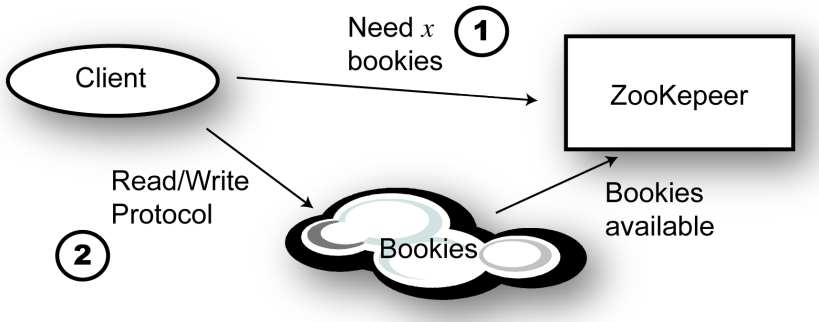
Guarantees

- Durability, confirm an add entry
  - ✓ Replicated on $f+1$ bookies
  - ✓ Synced to disk in every replica



Coordination and Metadata

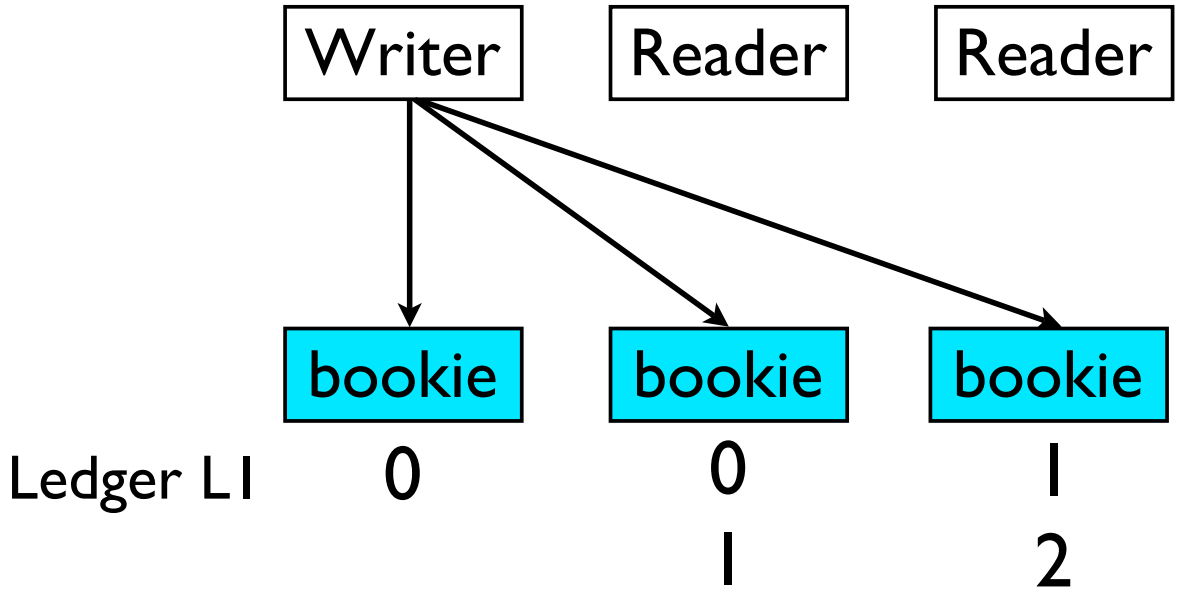
- Apache ZooKeeper
- Requirements
  - ✓ Ledger metadata
  - ✓ Available bookies
- Upon closing a ledger
  - ✓ Write id of the last confirmed entry



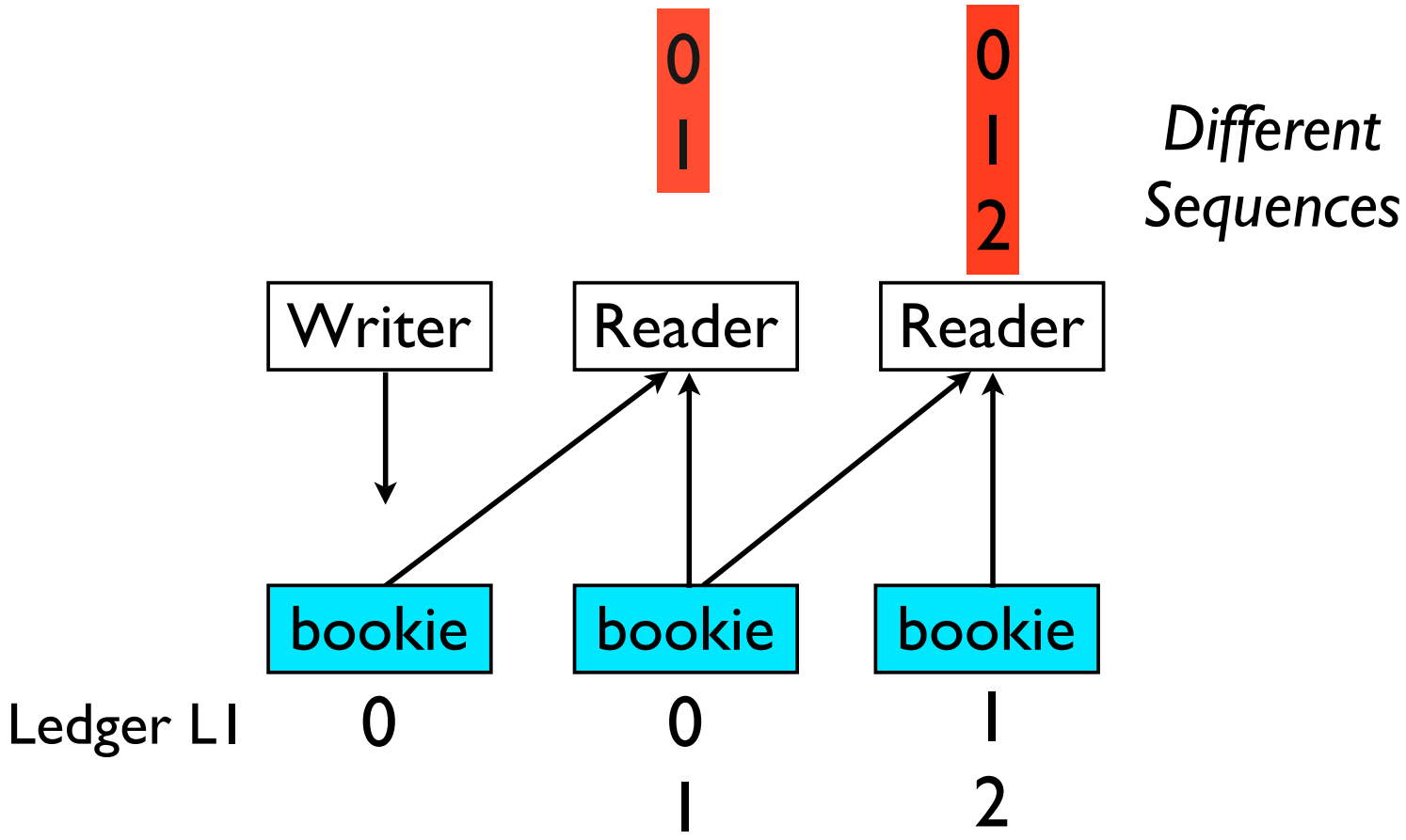
Why keep last entry id?

- Acknowledgement
  - ✓ Ledger closed properly
- Agreement
  - ✓ Two readers don't read different sets of entries

Why keep last entry id?



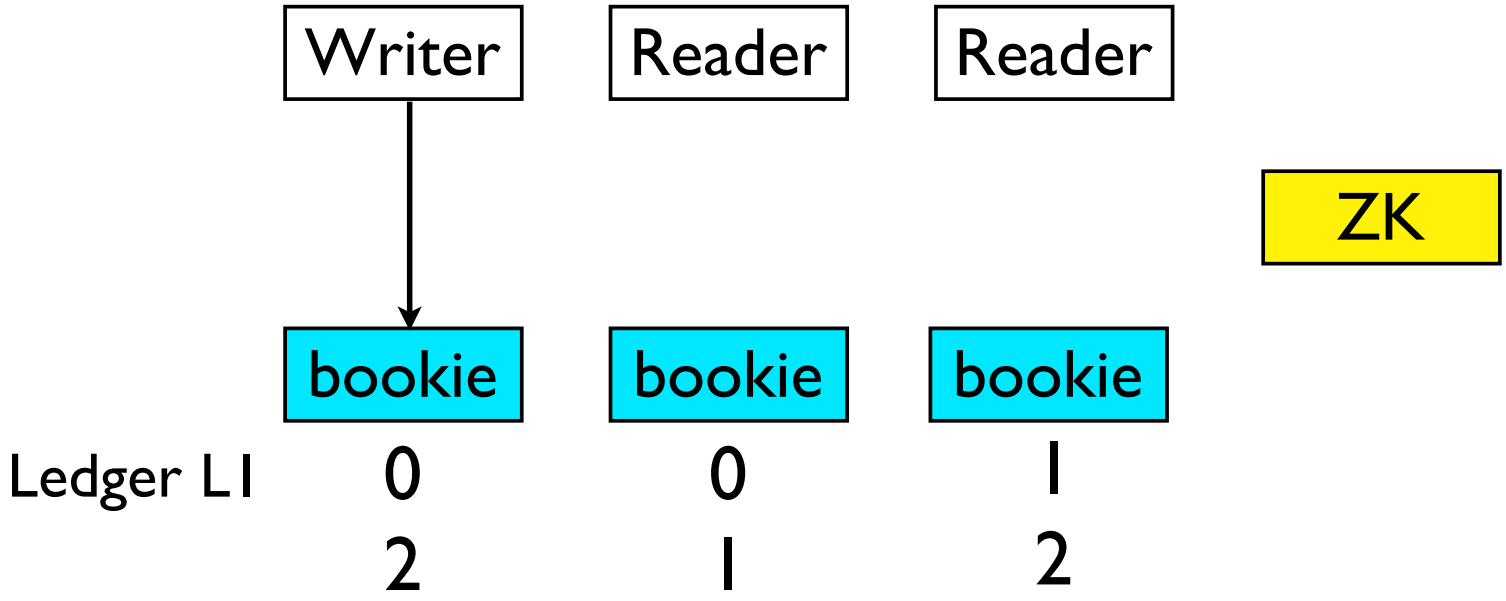
Why keep last entry id?



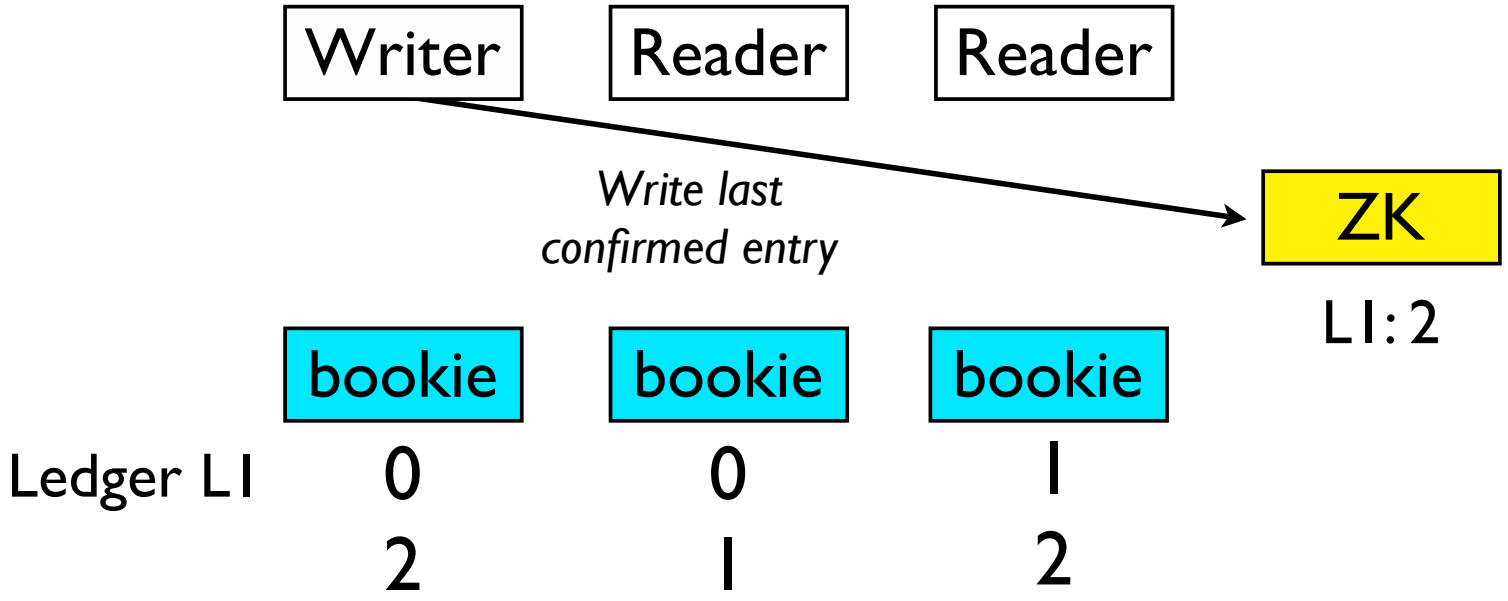
Why keep last entry id?

- Acknowledgement
  - ✓ Ledger closed properly
- Agreement
  - ✓ Two readers don't read different sets of entries
  - ✓ Consensus through ZooKeeper

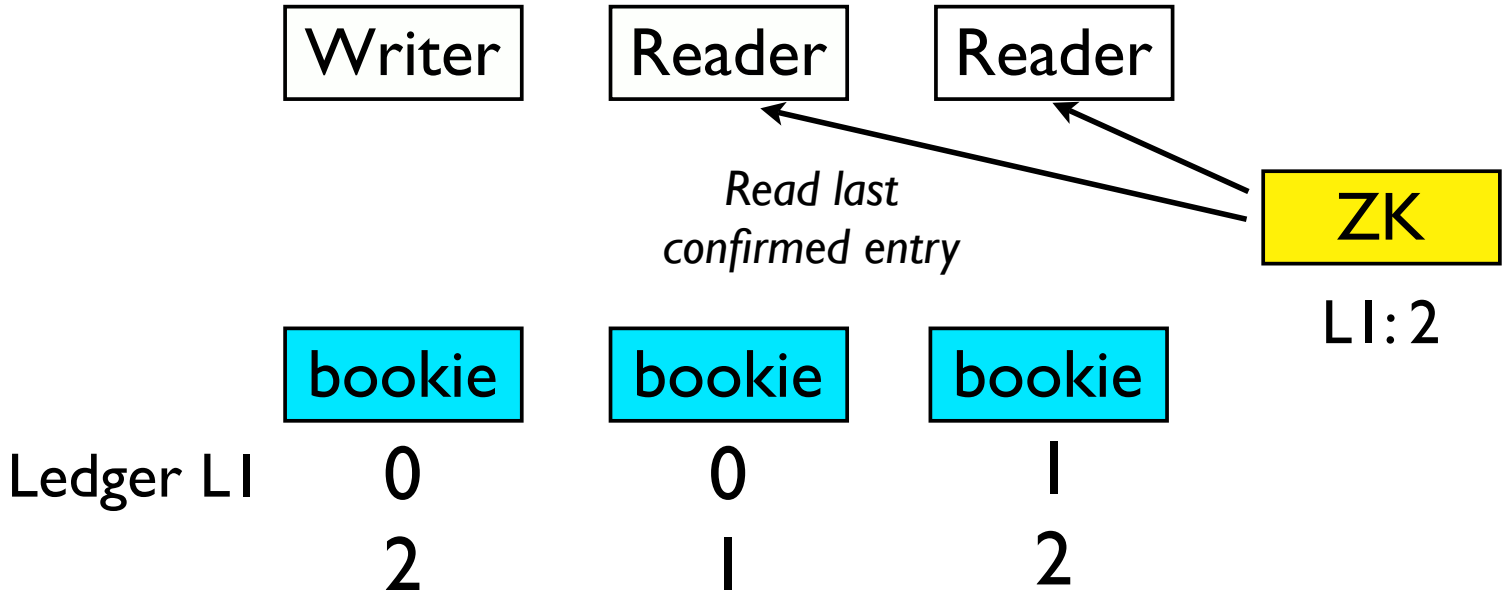
Why keep last entry id?



Why keep last entry id?



Why keep last entry id?



Guarantees

- Durability, upon acknowledgement
  - ✓ Replicated on $f+1$ bookies
  - ✓ Synced to disk in every replica
- Consistent reads, upon close
  - ✓ All readers read the same sequence

Why keep last entry id?

- Acknowledgement
  - ✓ Ledger closed properly
- Agreement
  - ✓ Two readers don't read different sets of entries
  - ✓ Consensus through ZooKeeper
- What if no last entry id has been written?

Recovery procedure

- Reader client executes a **ledger recovery** procedure
- Hints on ledger entries
- **Procedure**
  - ✓ Request last entry hint from bookies
  - ✓ Try to read as many entries greater than the hint
  - ✓ Make sure entries are written to a quorum

Scalability of writes

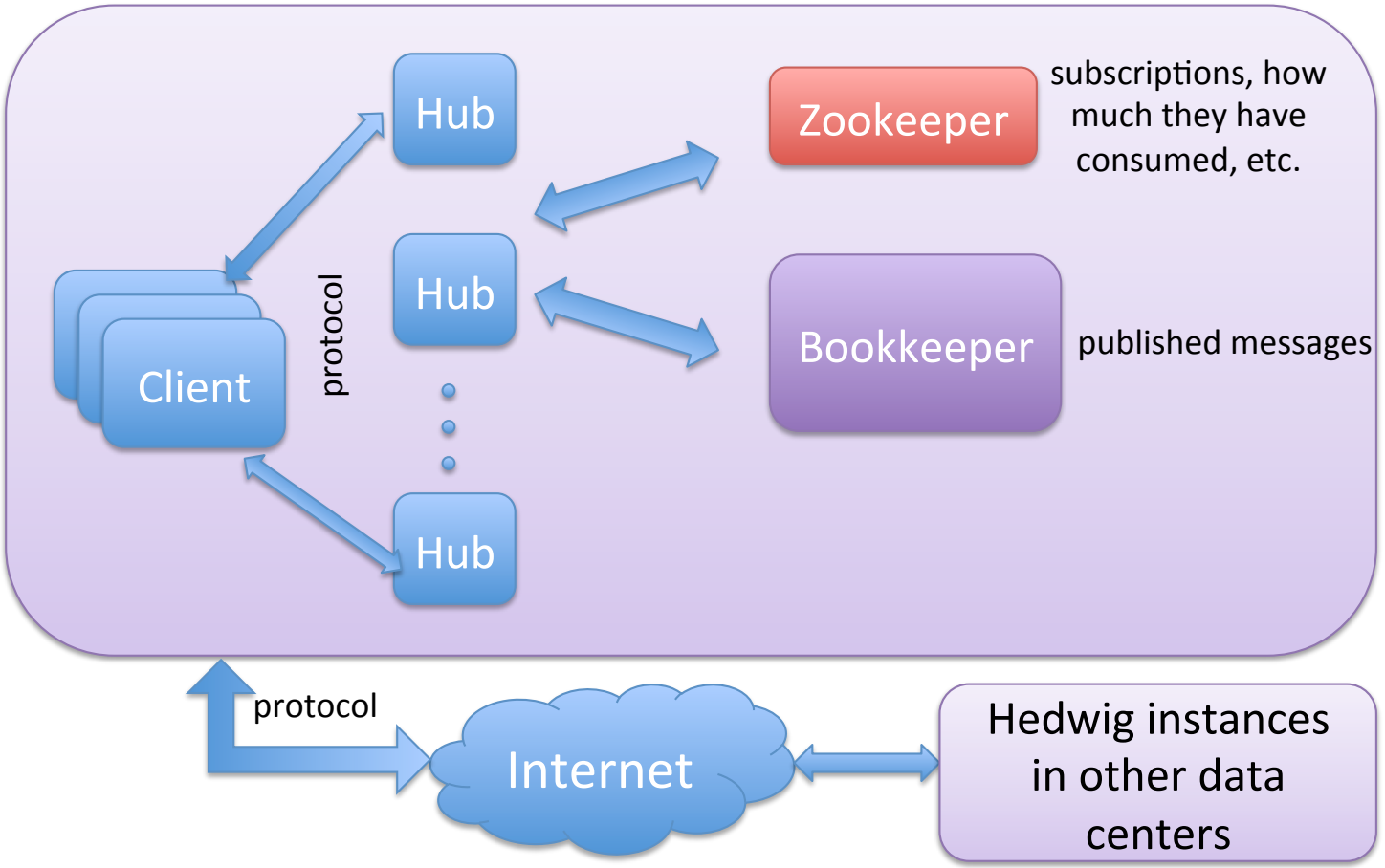
- Write quorums do not necessarily intersect
- Assuming that:
  1. Each bookie performs *e* entries/s
  2. Number of bookies: *r*
  3. Write quorum: *q* bookies
- Ideal maximum throughput: $\frac{r \times e}{q}$
- In practice, network bandwidth or cpu limits the total capacity in bytes written per second

Hedwig

Hedwig

- Multi-region pub/sub system
  - ✓ Multiple data centers
- Guaranteed-delivery topic-based pub-sub system
- Elastically scalable
  - ✓ Deployed over commodity machines
  - ✓ Capacity can be added on-the-fly by adding machines
- Low Operational Complexity
  - ✓ Tolerate failures without manual intervention
  - ✓ Automatic load balancing

Hedwig overview

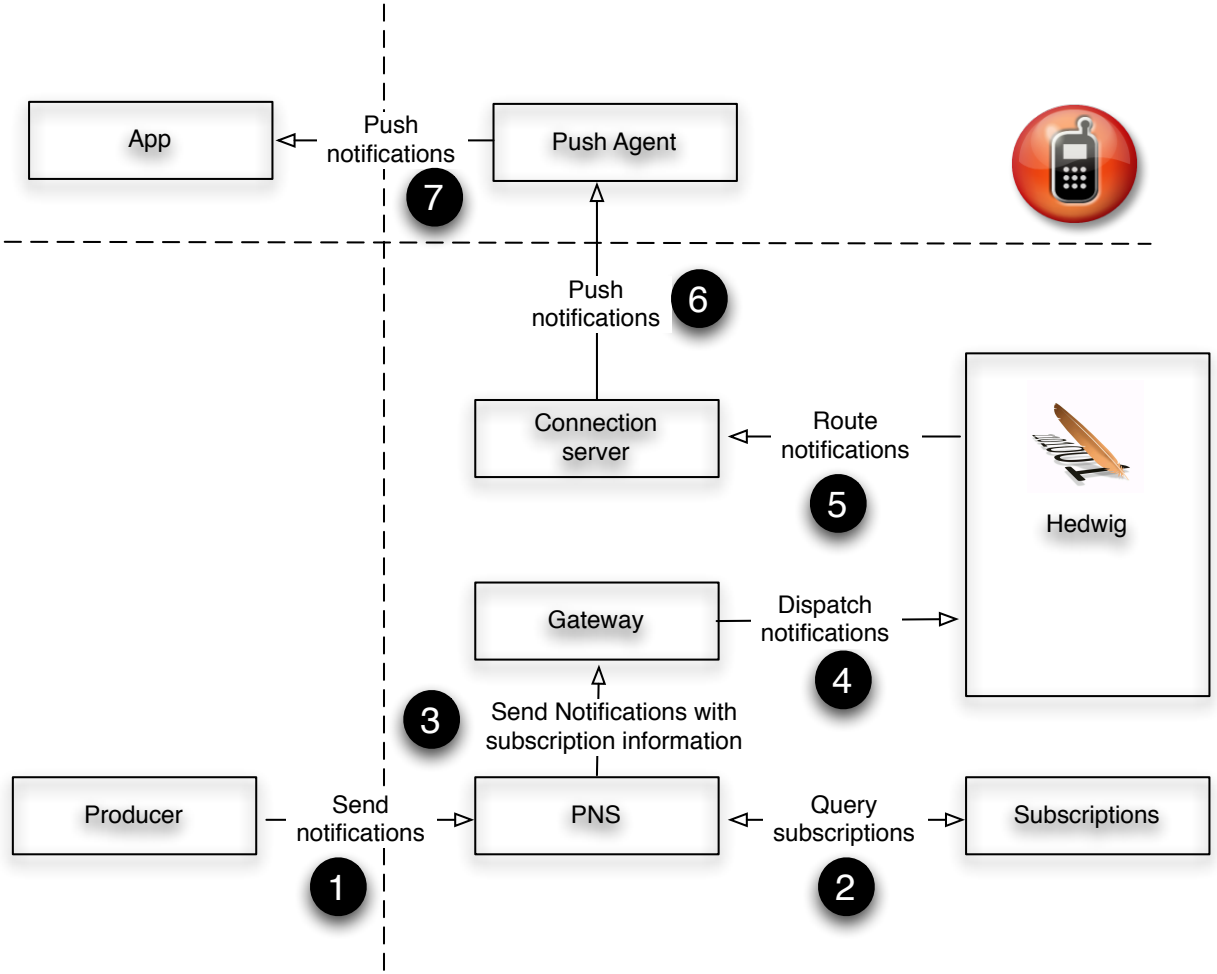


Use case

Push notifications

- Users of mobile devices
- Notifications
  - ✓ News alerts, social network updates, email, etc.
- Pushing is typically preferable over pulling
  - ✓ Lower latency
  - ✓ Saves on battery

Push notifications



Designed to serve tens to hundreds of millions of users

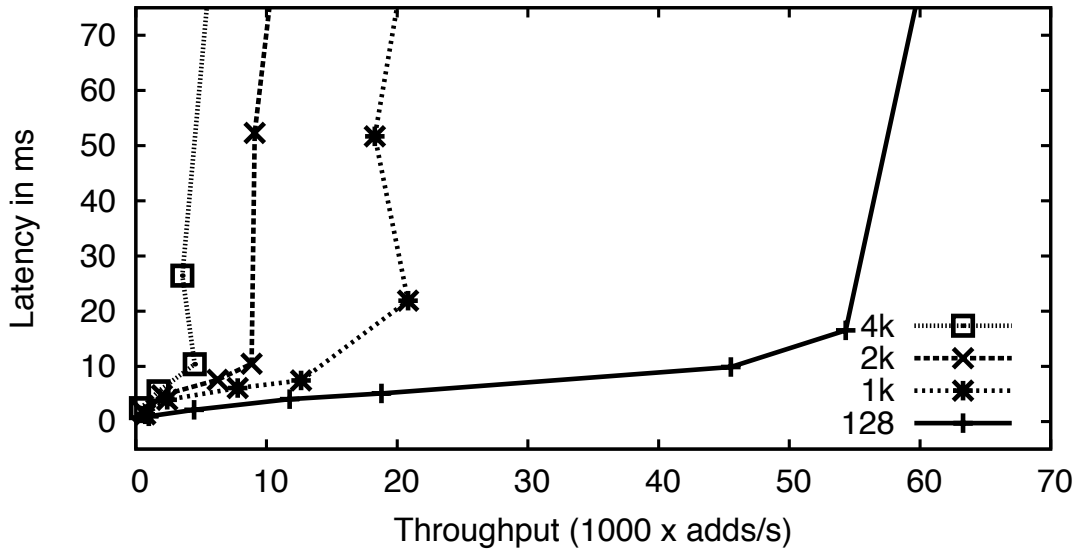
Performance

Setup

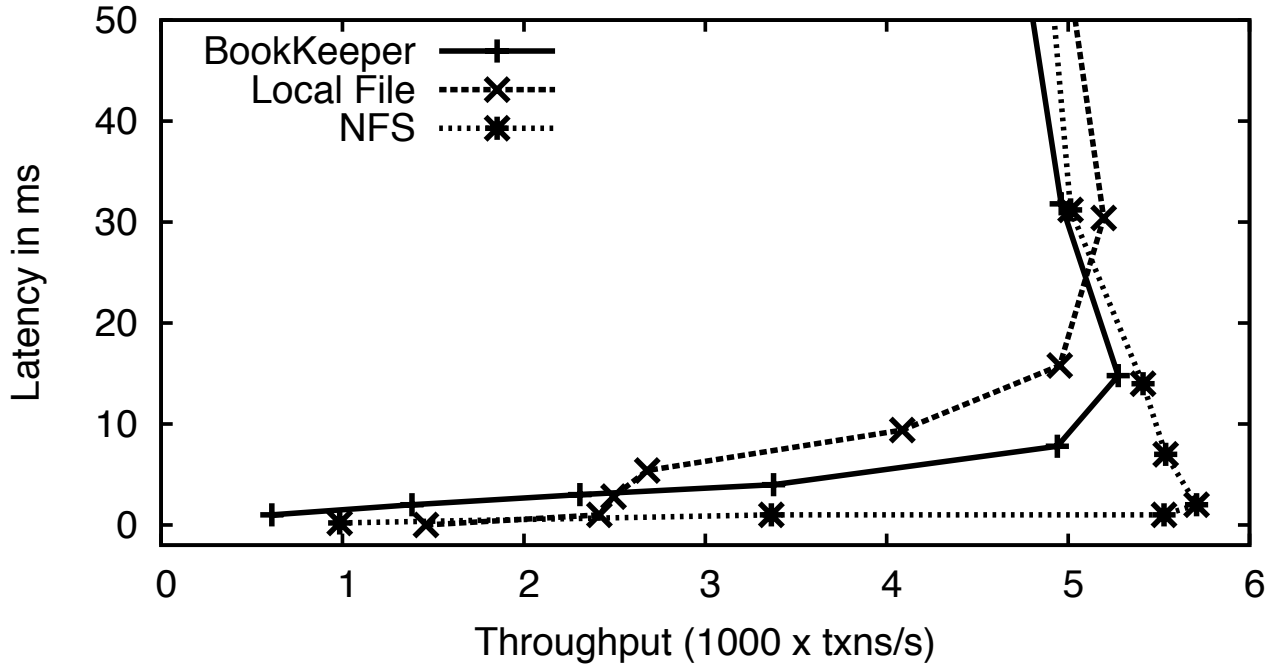
- Cluster of identical machines
- 2 Quad Core Intel Xeon 2.5GHz
- 16GB of RAM
- Four SATA disks, 7,200 RPMs
- 1Gbit/s network interface

BookKeeper performance

- Single writer



BookKeeper and the Namenode



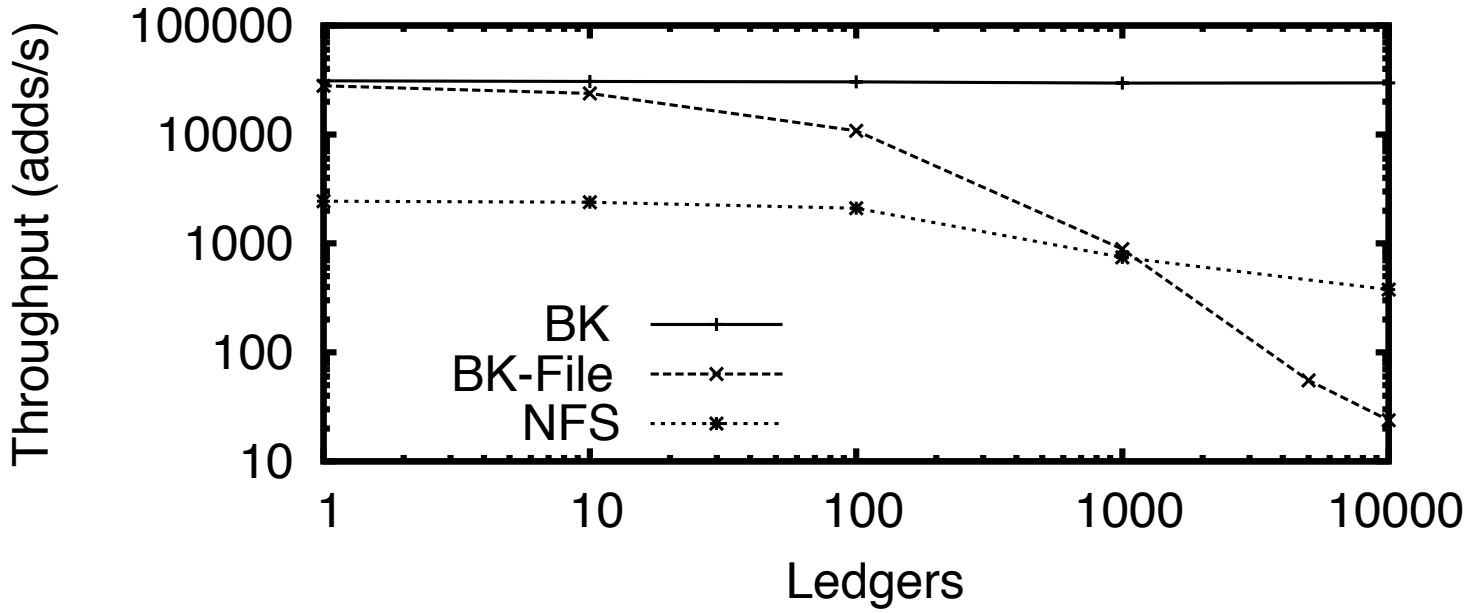
BookKeeper performance

- Multi-writer
  - ✓ Aggregate throughput

bytes	2Q		3Q	
	3E	6E	3E	6E
128	107k	197K	81k	138k
1024	31k	58k	20k	35k
4096	7k	16k	4k	9k

add operations/s

Concurrent ledgers



Wrap up

Advanced features

- Auto-recovery
  - ✓ Replicate upon crashes
- Metadata store
  - ✓ Large deployments need a lot of metadata
  - ✓ Originally ZooKeeper
- Managed Ledgers

Status

- Apache release 4.1.0
- BookKeeper and the namenode
  - ✓ Code available in trunk and Hadoop version 2

Team

- Active committers
  - ✓ Sijie Guo (Yahoo!)
  - ✓ Flavio Junqueira (Yahoo! Research)
  - ✓ Ivan Kelly (Yahoo! Research)
- Other important contributions
  - ✓ Cloud Messaging at Yahoo!
  - ✓ Huawei (Uma Maheswara, Rakesh)
  - ✓ Twitter

Conclusion

- Durability
  - ✓ Efficient writes to stable storage
  - ✓ Concurrent writes from distinct processes
- BookKeeper
  - ✓ Shared storage for logs
  - ✓ Open source and in production

Questions?

http://zookeeper.apache.org/bookkeeper