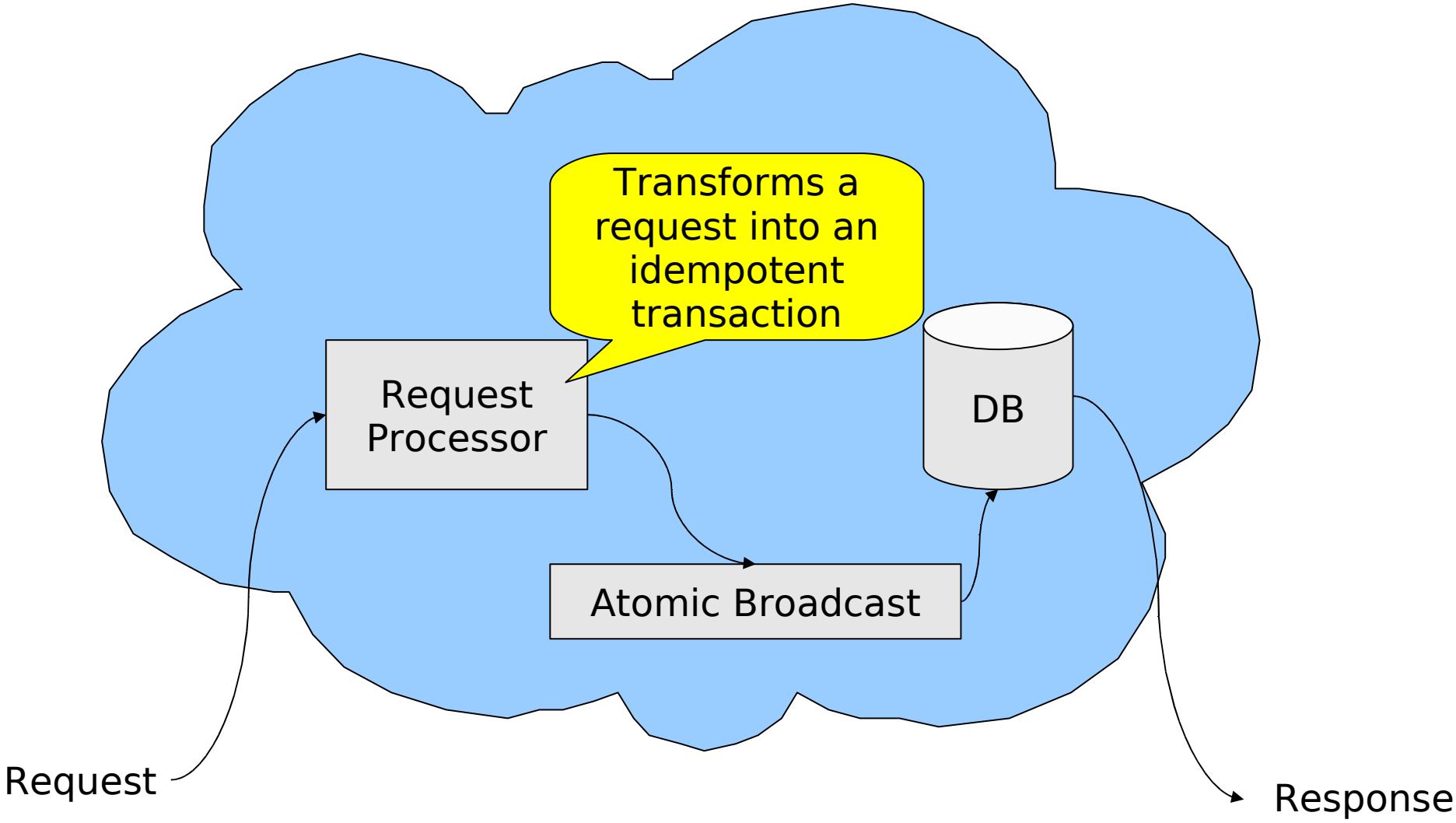
ZooKeeper Atomic Broadcast

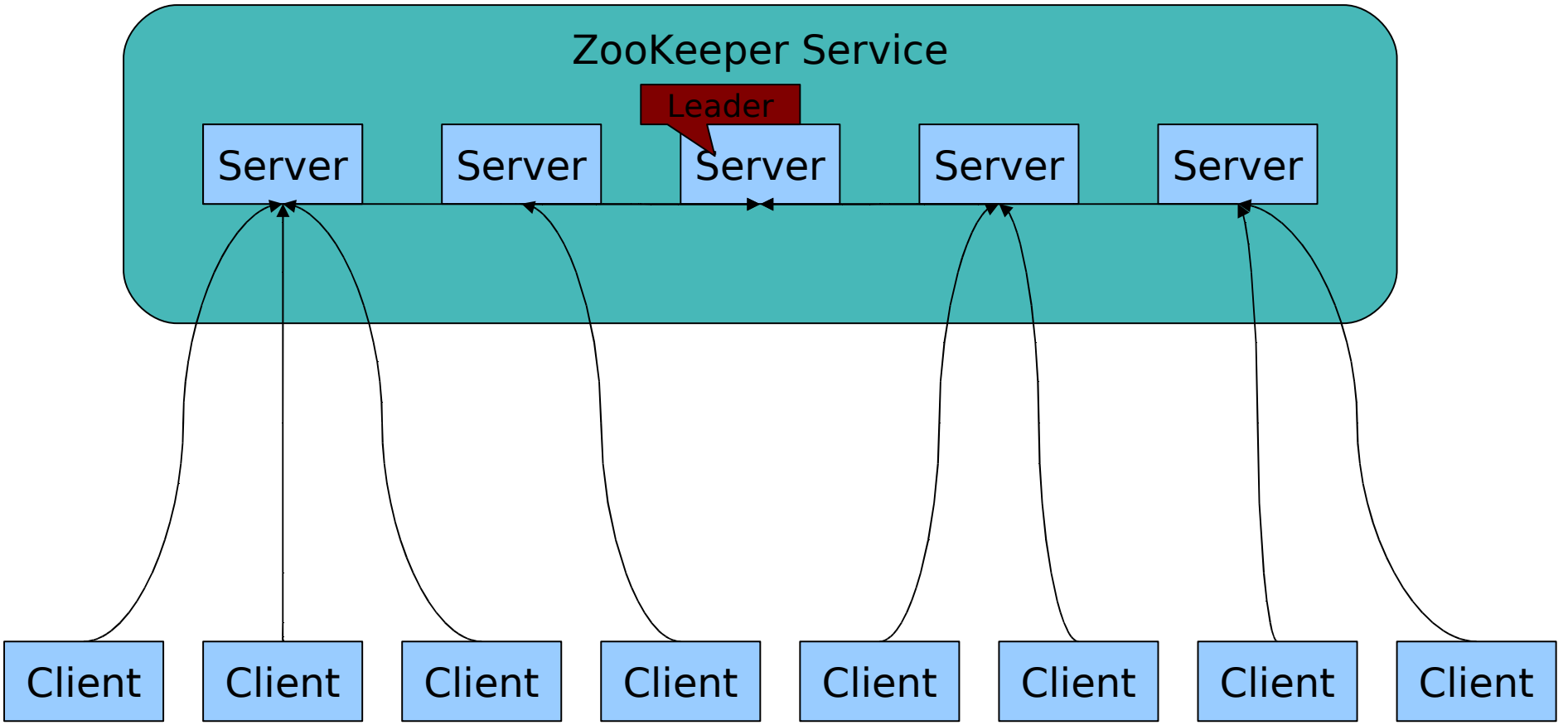
The heart of the ZooKeeper coordination service

Benjamin Reed, Flavio Junqueira
Yahoo! Research

ZooKeeper Service



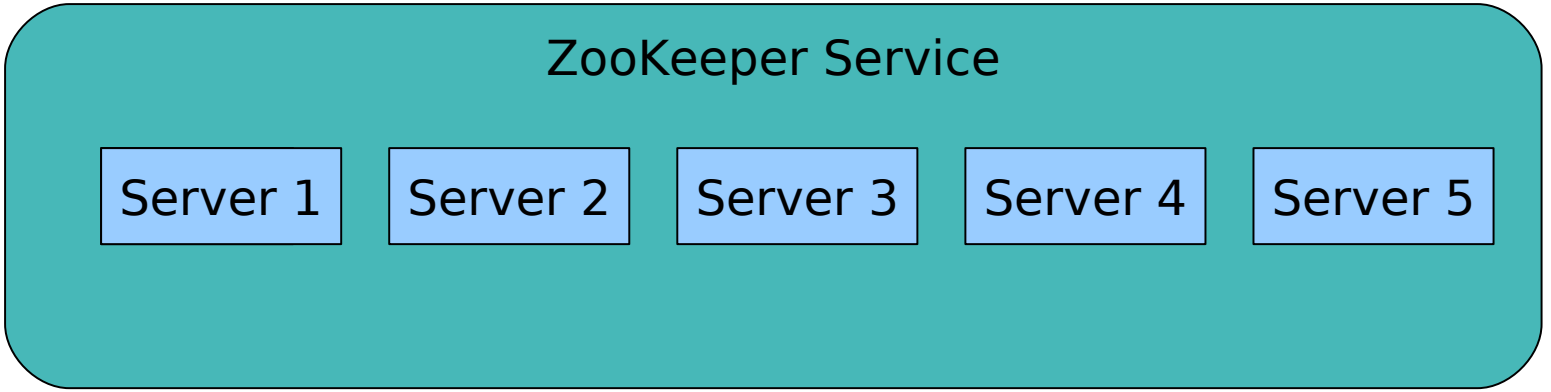
ZooKeeper Servers



Goals

1) Must be able to tolerate failures
2) Must be able to recover from correlated recoverable failures (power outages)
3) Must be correct
4) **Must be easy to implement correctly**
5) Must be fast (high throughput, low latency)
   - Bursty throughput
   - Homogeneous servers with non homogeneous behavior (some will inevitably be faster than others because of HW or runaway processes etc)

ZooKeeper Leader Election

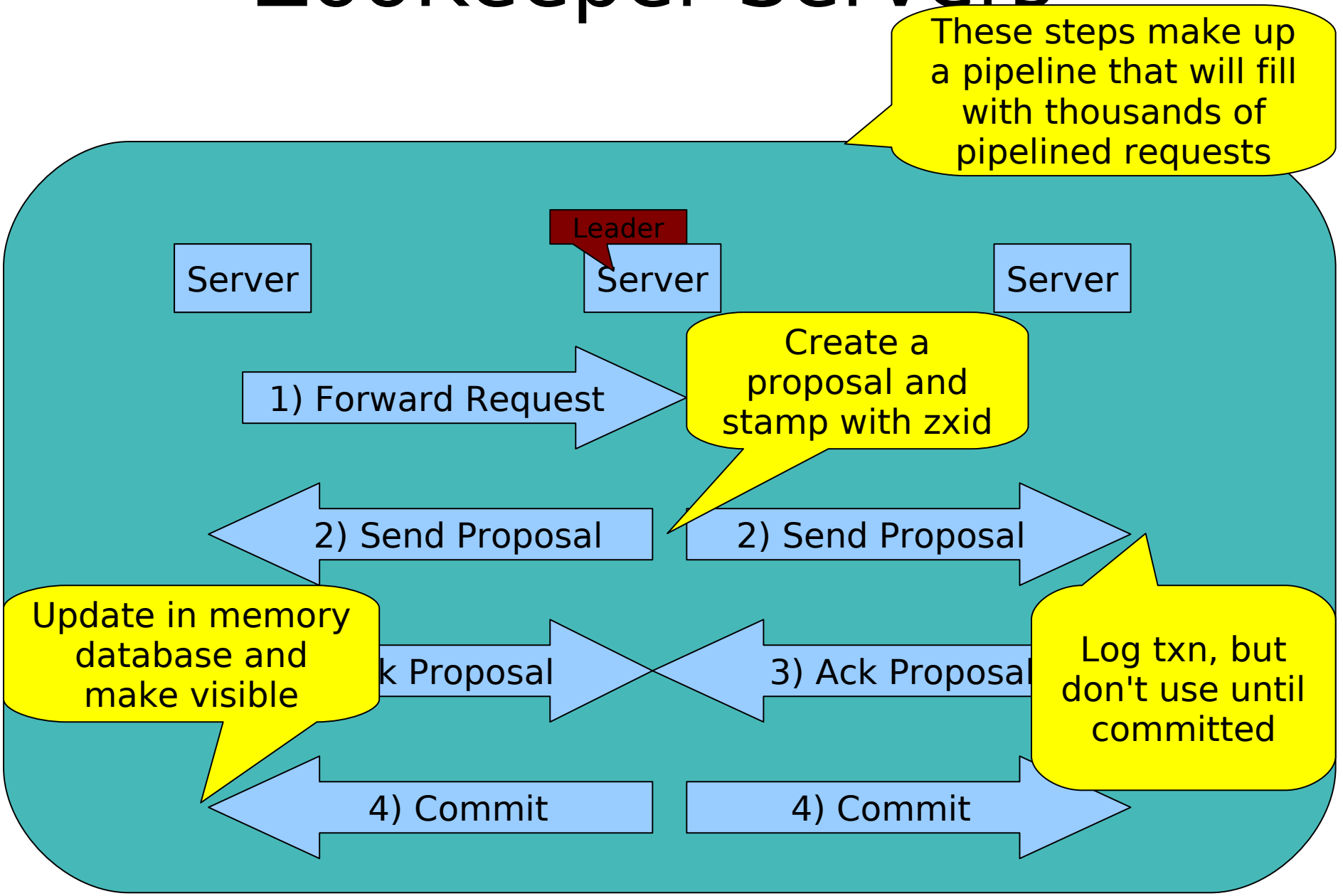


1) UDP or TCP based
2) Server with the highest logged transaction gets nominated
3) Election doesn't have to be absolutely successful, just very likely successful

Starting assumption

1) Ability to create FIFO channels
   - We use TCP
   - Theoretically not a stronger assumption than classic lossy unordered channel since that is what TCP is built on
2) Crash fail
   - Digests to detect corruption
3) 2f+1 servers to handle f failures
   - Service must be able to recover from correlated recoverable failures (power outages)

ZooKeeper Servers



Nice Properties

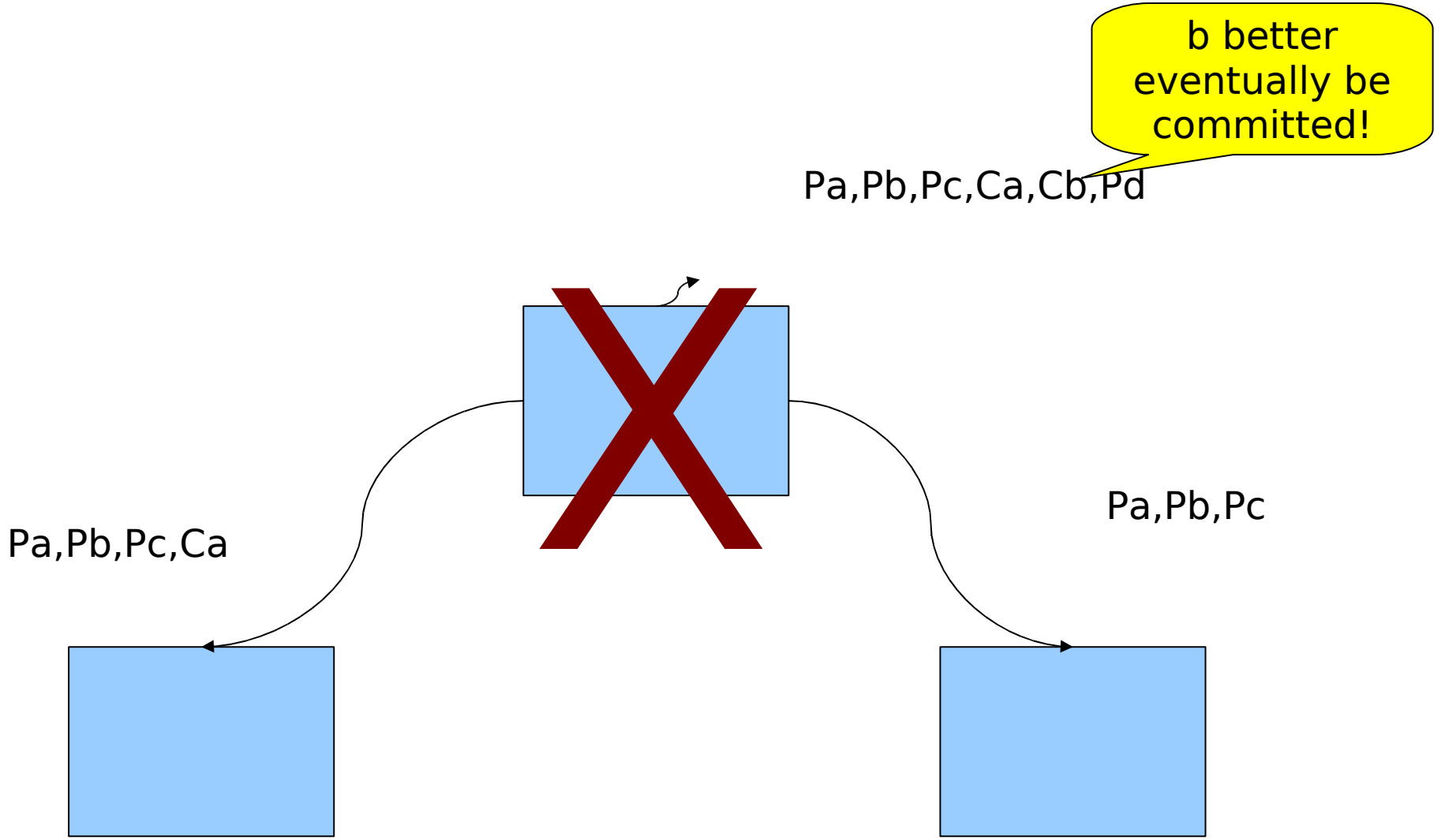
1) Leader always proposes in order
2) Because we use TCP, followers always receive in order
3) Followers process proposals in order
4) TCP means that Leader will get ACKs in order and thus commit in order
5) Followers only need to connect to a single server
6) Leader just waits for connections

Everything is cool until...

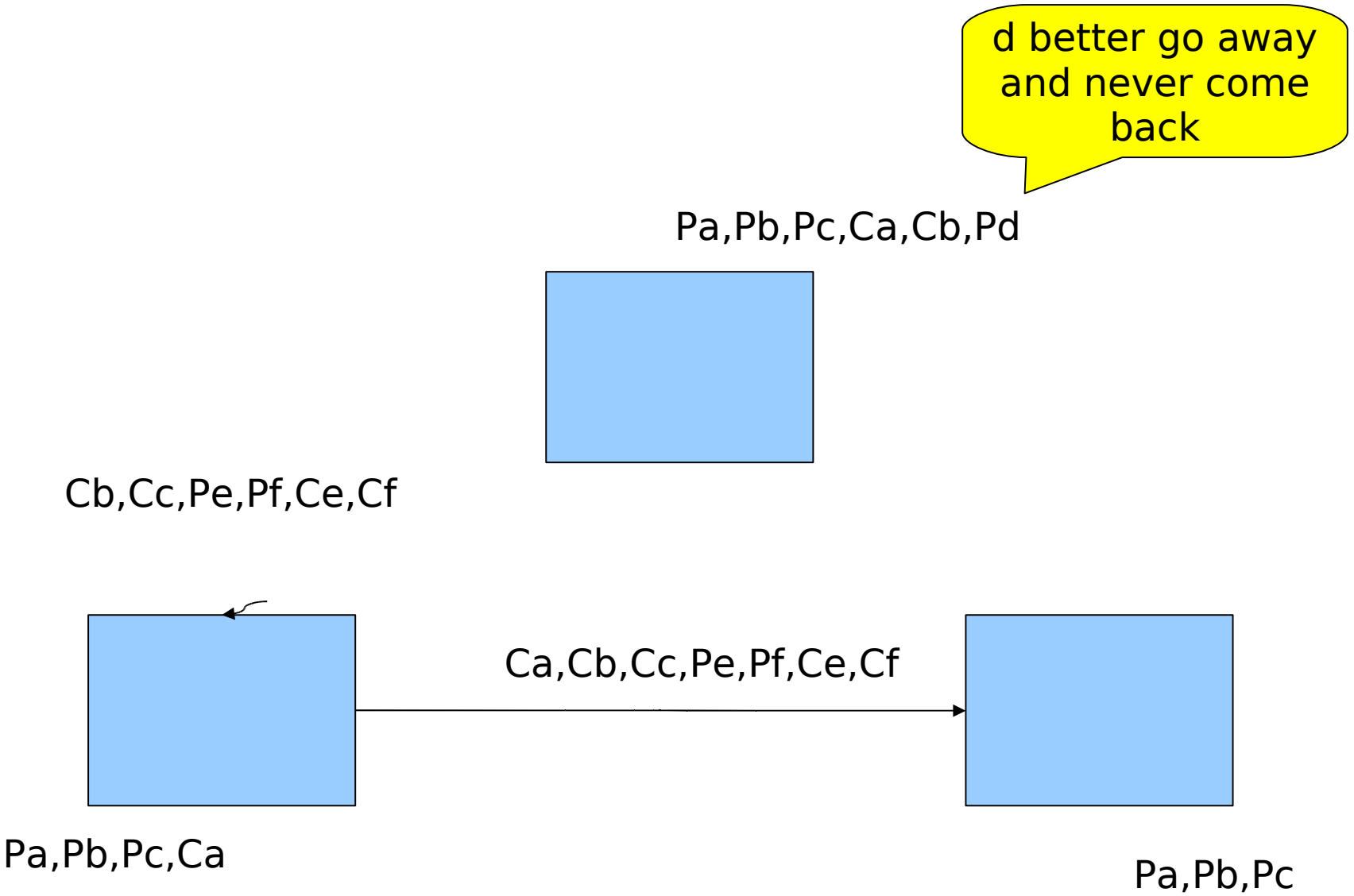
Leader Failure!

2) Make sure that the what has been delivered to some get delivered to all
3) Make sure that what gets forgotten stays forgotten
4) We get to choose what to do with the stuff in between

Missed deliveries



Bad Recall

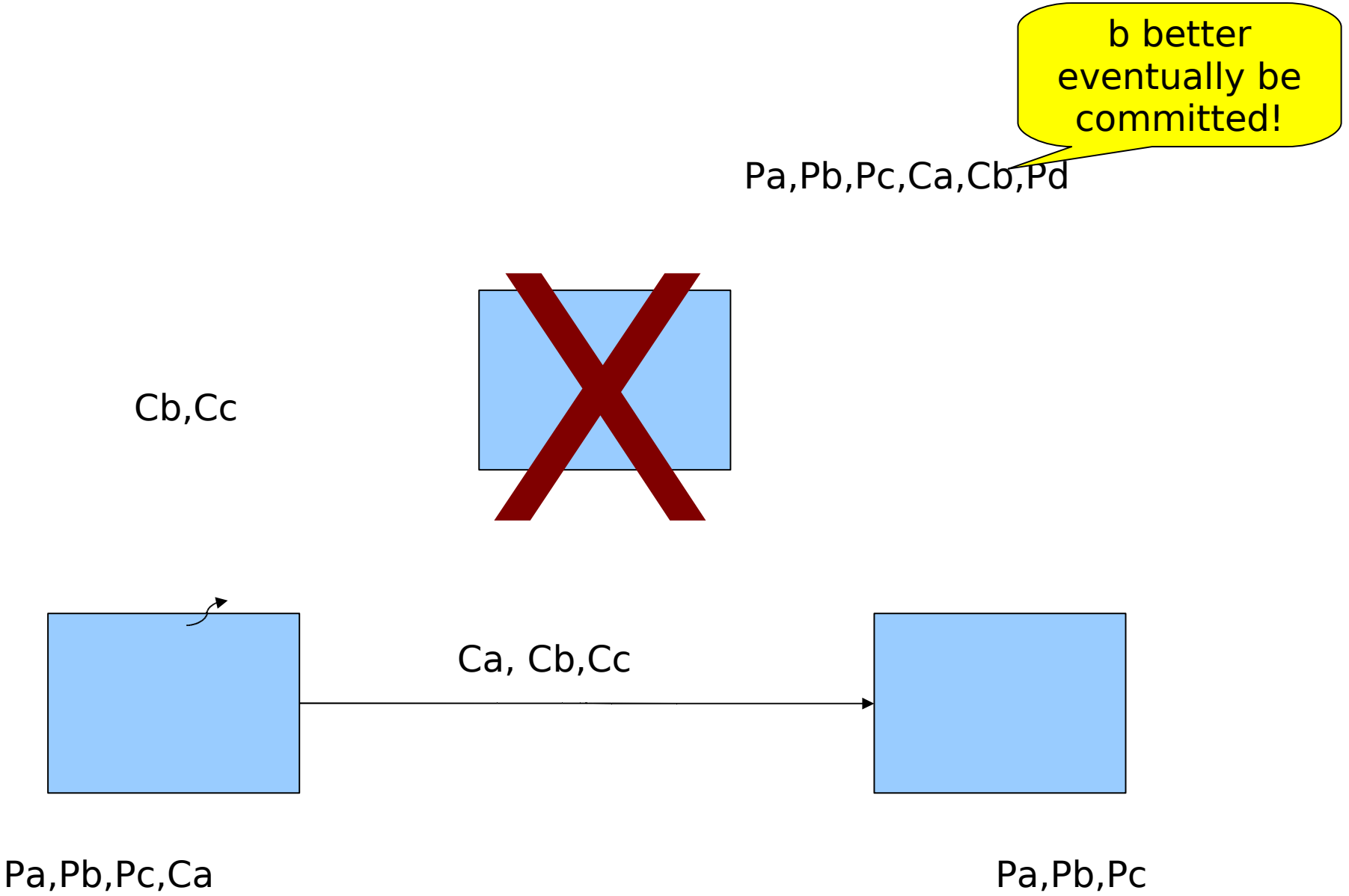

d better go away and never come back
Pa,Pb,Pc,Ca,Cb,Pd
Cb,Cc,Pe,Pf,Ce,Cf
Ca,Cb,Cc,Pe,Pf,Ce,Cf
Pa,Pb,Pc,Ca
Pa,Pb,Pc

Never forget

1) If we elect the right guy, we will not forget anything

- A new leader is elected by a quorum of followers
- Committed messages must be seen by at least someone in the quorum
- Elect the server that has seen the highest message in a quorum
- New leader will commit all proposals it has seen from the previous leader

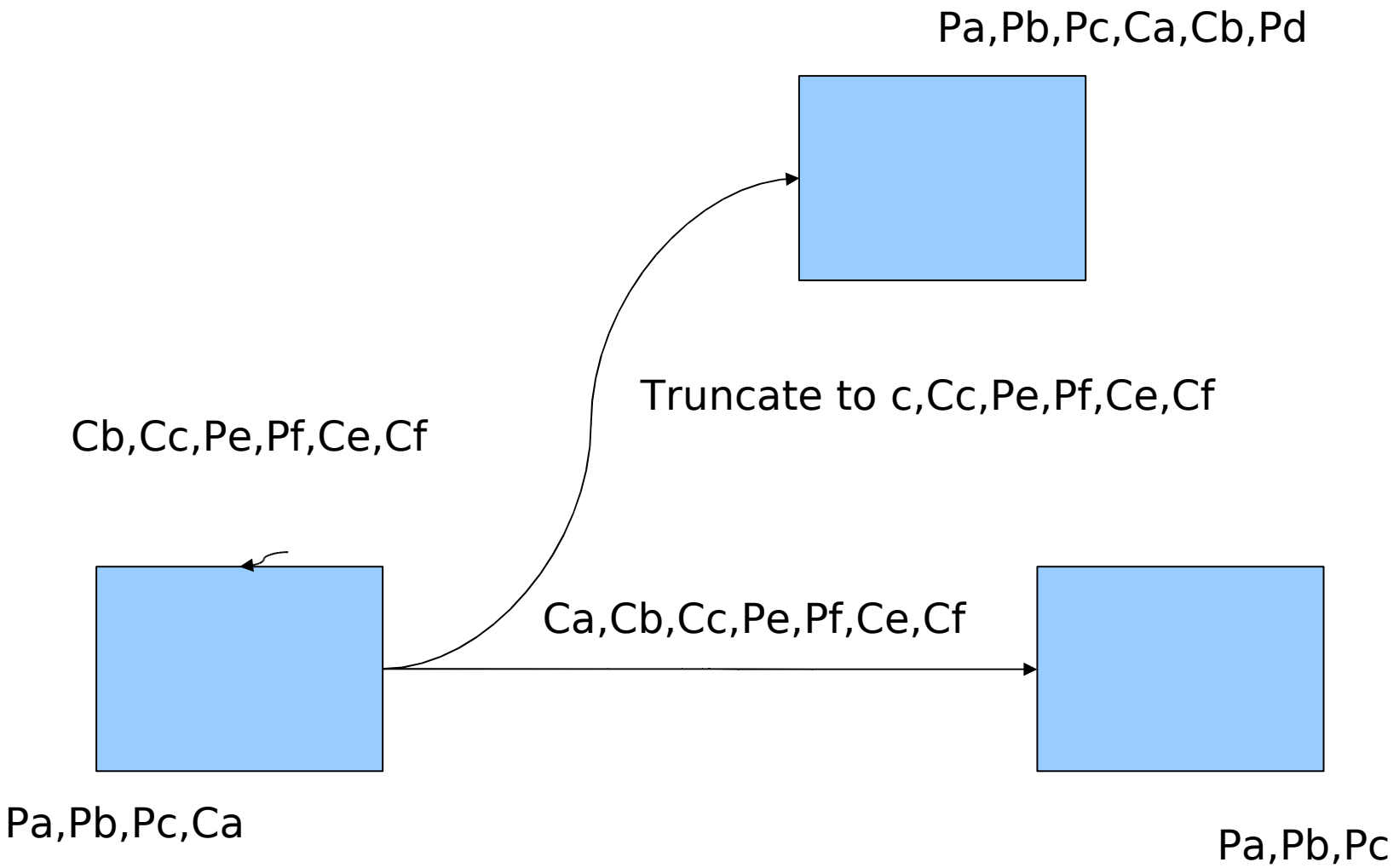
Missed deliveries



Letting go

1) We use epochs to make sure that we only recover the last leaders outstanding proposals once.
   - Zxid is a 64-bit number: 32-bit of epoch and 32-bit counter
   - A new leader will increment the epoch
   - A new leader will only start proposing once the previous epoch is cleaned

Bad Recall


Pa,Pb,Pc,Ca,Cb,Pd
Truncate to c,Cc,Pe,Pf,Ce,Cf
Cb,Cc,Pe,Pf,Ce,Cf
Ca,Cb,Cc,Pe,Pf,Ce,Cf
Pa,Pb,Pc,Ca
Pa,Pb,Pc

Leader Protocol in a nutshell

1)At startup wait for a quorum of followers to connect

2)Sync with a quorum of followers

- Tell the follower to delete any txn that the leader doesn't have (easy since it will only differ in one epoch)
- Send any txns that the follower doesn't have

3)Continually

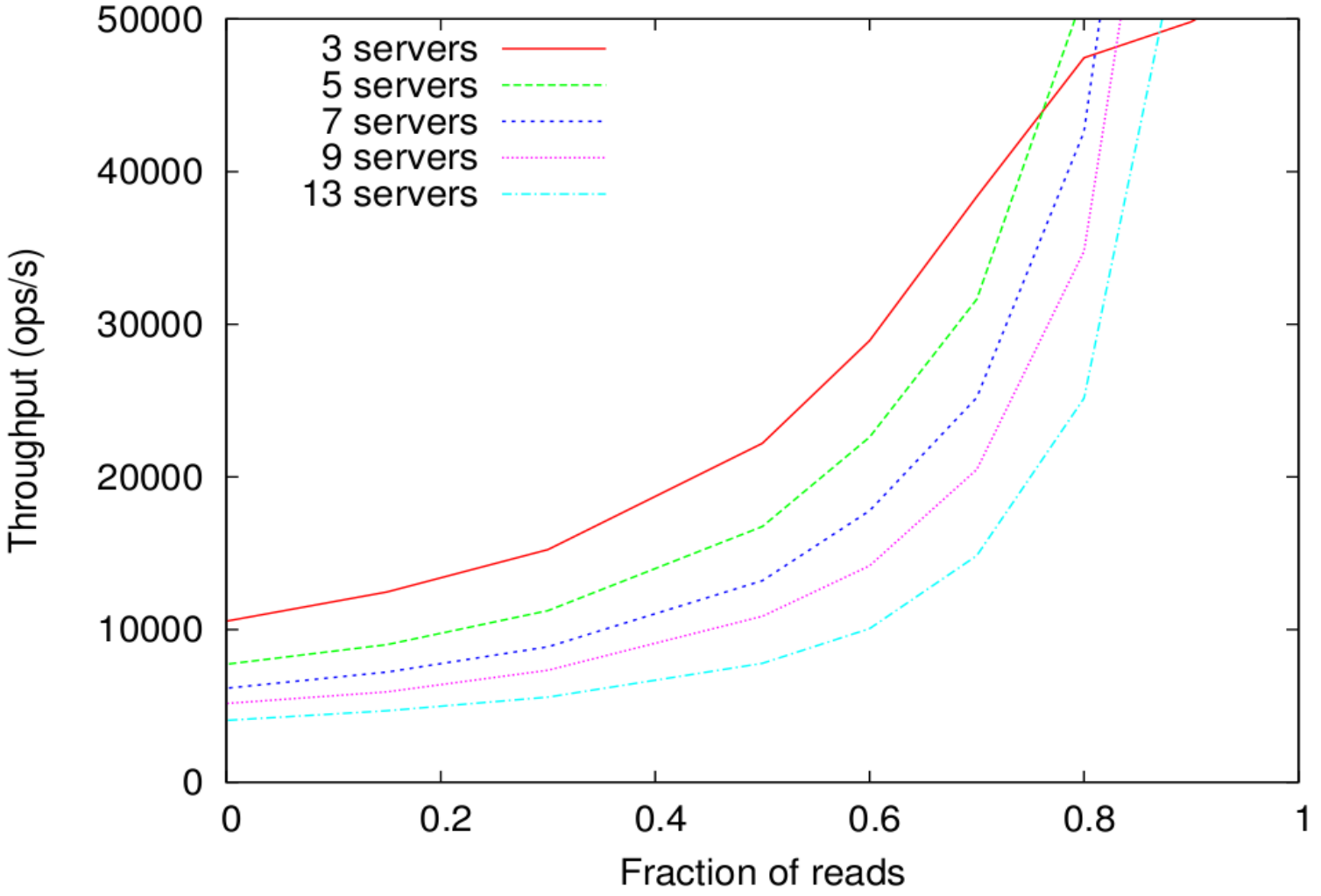
- Assign and zxid to any message to be proposed and broadcast proposals to followers
- When a quorum has acked a proposal broadcast a commit

(Broadcast means queue the message to the TCP channel of each follower)

Follower protocol in a nutshell

1) Connect to a leader
2) Delete any txns in the txn log that the leader says to delete
3) Continually
   - Log to the txn log proposed transactions and send an ack to leader
   - Deliver any committed txn

Performance



Status

1) An Apache project http://hadoop.apache.org/zookeeper
2) Used extensively at Yahoo! Also used by non Yahoo! Projects
3) Future work:
   - Observers
   - Tree distribution network