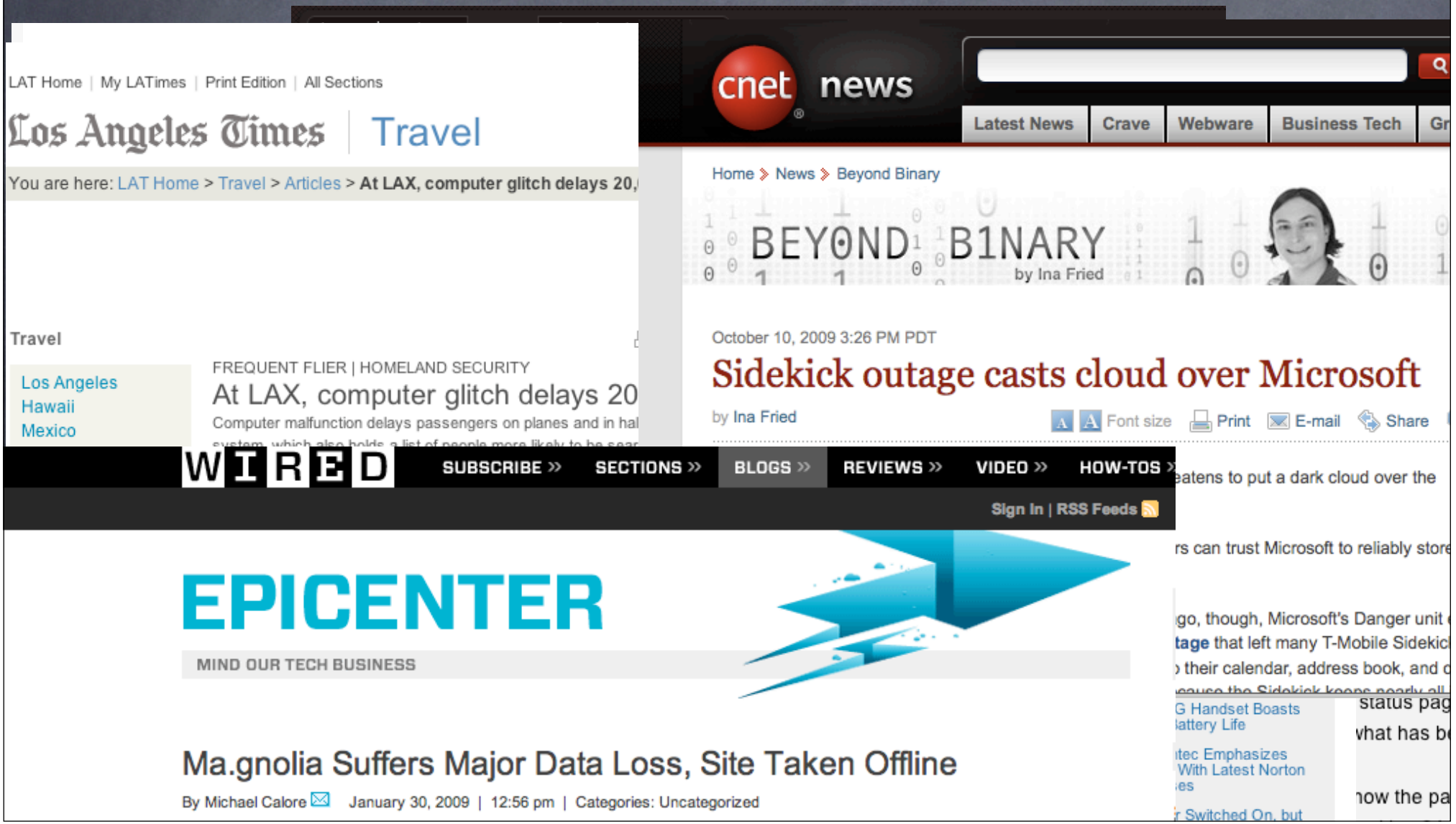
UpRight Cluster Services

Allen Clement, Manos Kapritsos,
Sangmin Lee, Yang Wang, Lorenzo Alvisi,
Mike Dahlin, Taylor Riche

The University of Texas at Austin

Failures are not fail-stop



To the rescue

- Byzantine Fault Tolerance (BFT)
  - tolerate f arbitrary failures
    - safe always
    - good performance with failures if network behaves well
    - eventually live

This talk

- BFT in real systems
  - ZooKeeper, **H**adoop **D**istributed **F**ile **S**ystem
- What does it take?
  - Revising much of what we think we know
    - Failure model
    - BFT implementation
    - API

This talk

- BFT in real systems
  - ZooKeeper, **H**adoop **D**istributed **F**ile **S**ystem
- What does it take?
  - Revising much of what we think we know
    - Failure model
    - BFT implementation
    - API

For better or for worse

- Byzantine model is most general
  - all you need are $3f+1$ replicas...

Up Right

Up

- $u =$ maximum number of failures under which liveness* is ensured

Right

- $r =$ maximum number of malicious failures under which safety is preserved

Up Right

- $u =$ maximum number of failures under which liveness* is ensured
- $r =$ maximum number of malicious failures under which safety is preserved

(Lamport 2003; Dutta et al 2005; El-Malek et al 2005)

Up Right

- $u =$ maximum number of failures under which liveness* is ensured
- $r =$ maximum number of malicious failures under which safety is preserved

(Lamport 2003; Dutta et al 2005; El-Malek et al 2005)

- agreement : $2u+r+1$ replicas

Up Right

- $u =$ maximum number of failures under which liveness* is ensured
- $r =$ maximum number of malicious failures under which safety is preserved

(Lamport 2003; Dutta et al 2005; El-Malek et al 2005)

Replicas required for agreement

$2u+r+1$

	u=0	u=1	u=2	u=3
r=0	1	3	5	7
r=1	2	4	6	8
r=2	3	5	7	9
r=3	4	6	8	10

Crash tolerant

"Pay per B" FT

BFT

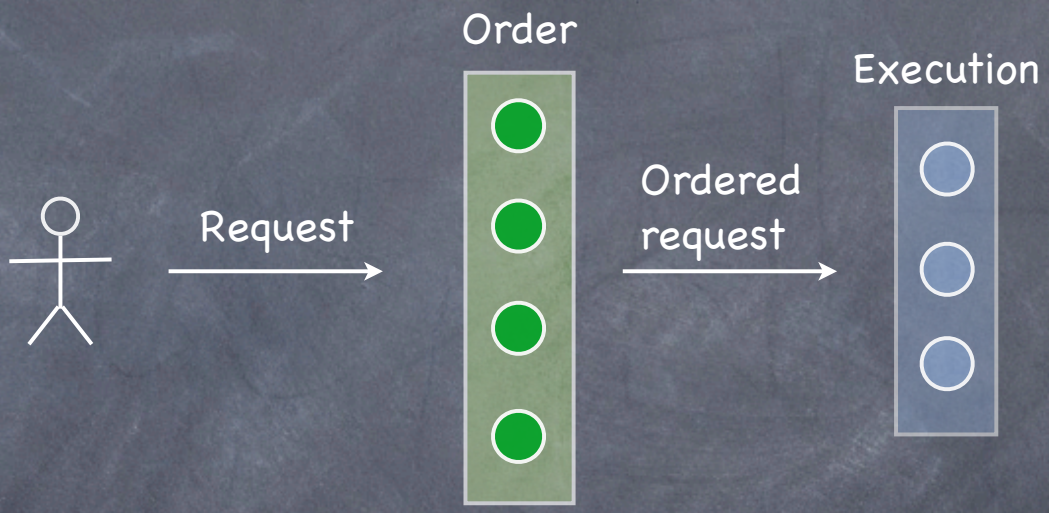
One Library to Rule Them All

- Only pay for the fault tolerance you need
- One fault tolerant library

Revising what we think we know

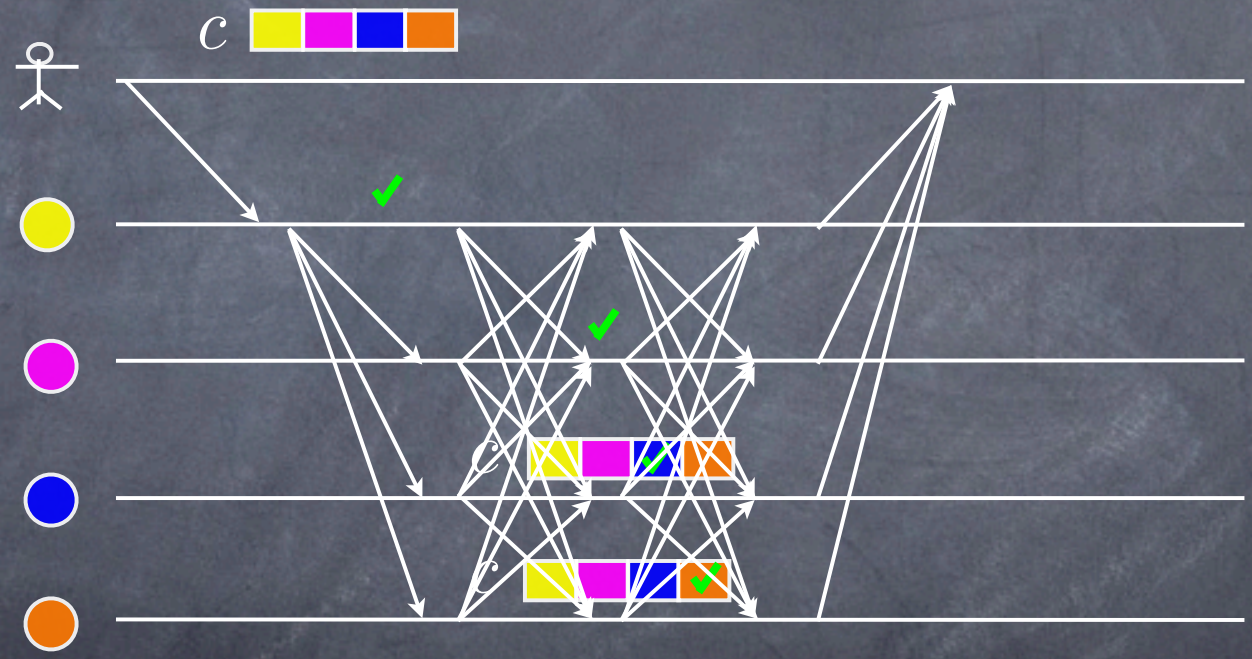
- Failure model
- BFT implementation
- API

Separating Order from Execution



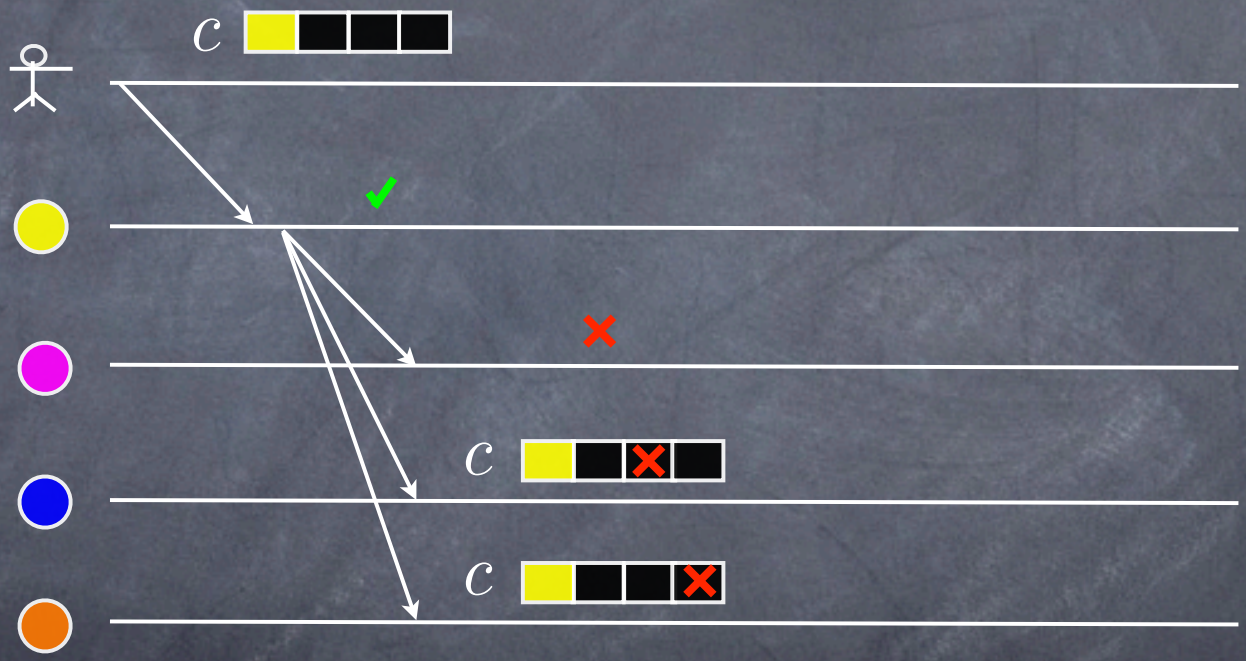
- Separating agreement from execution for Byzantine fault tolerant services [SOSP 2003]

Big MAC Attack



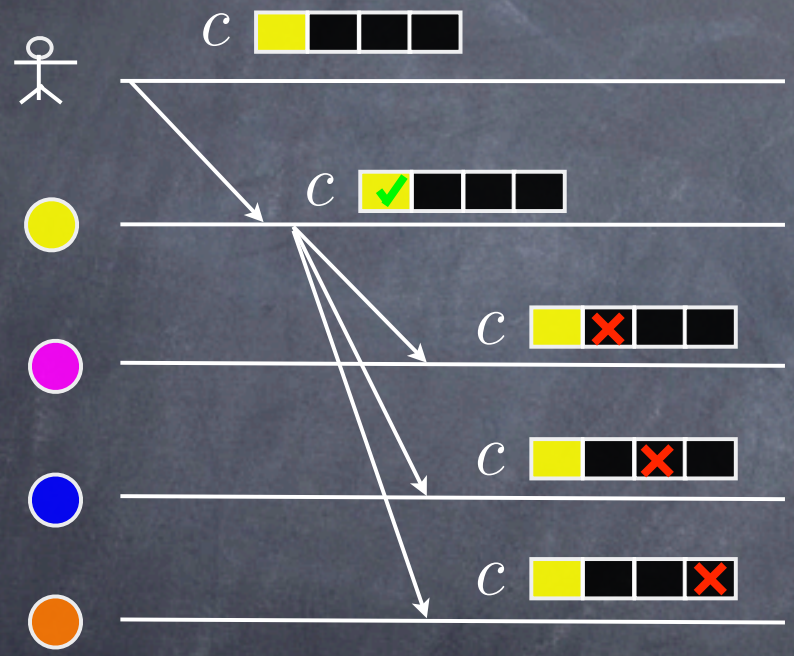
- Making BFT systems tolerate Byzantine failures [NSDI 2009]

Big MAC Attack

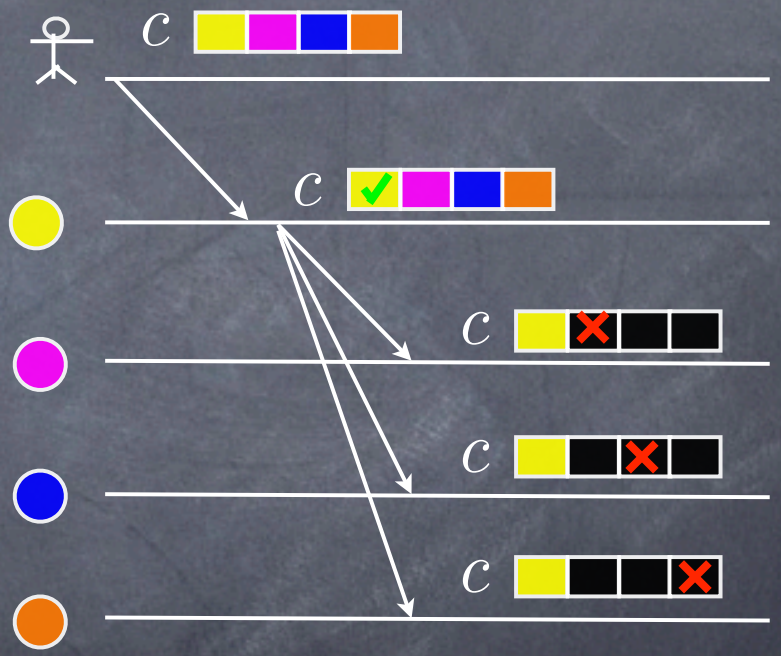


- Making BFT systems tolerate Byzantine failures [NSDI 2009]

Big MAC Attack

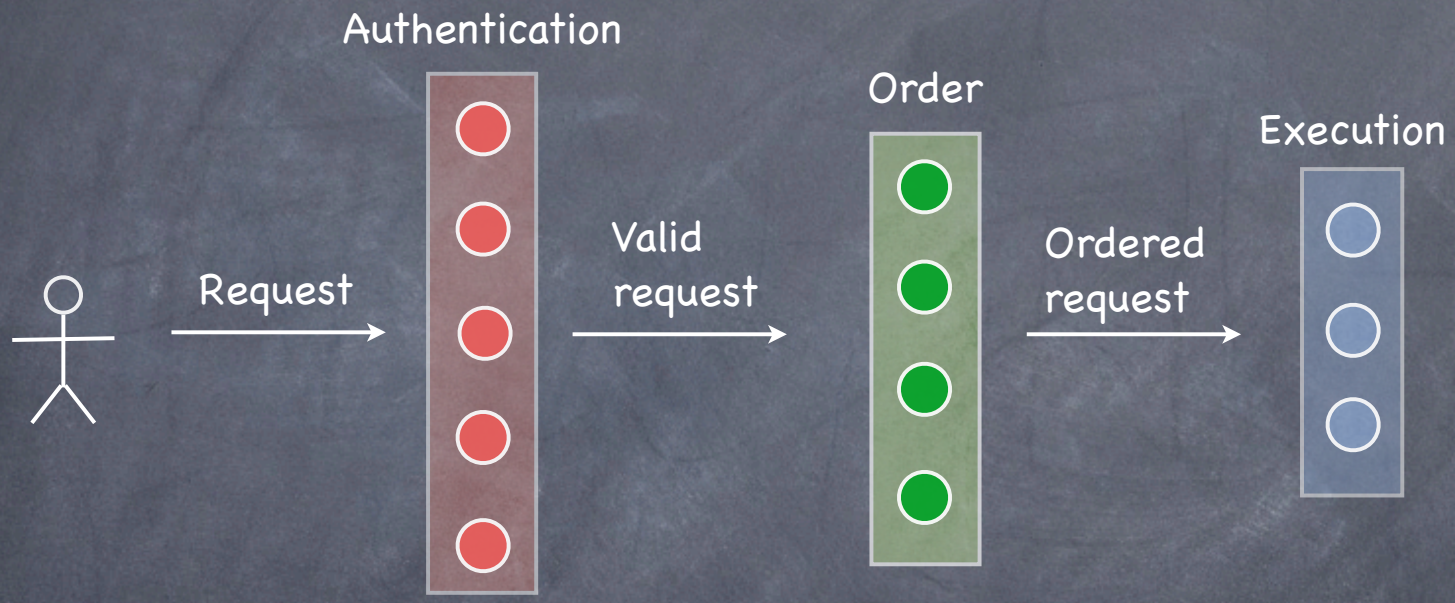


Faulty Client

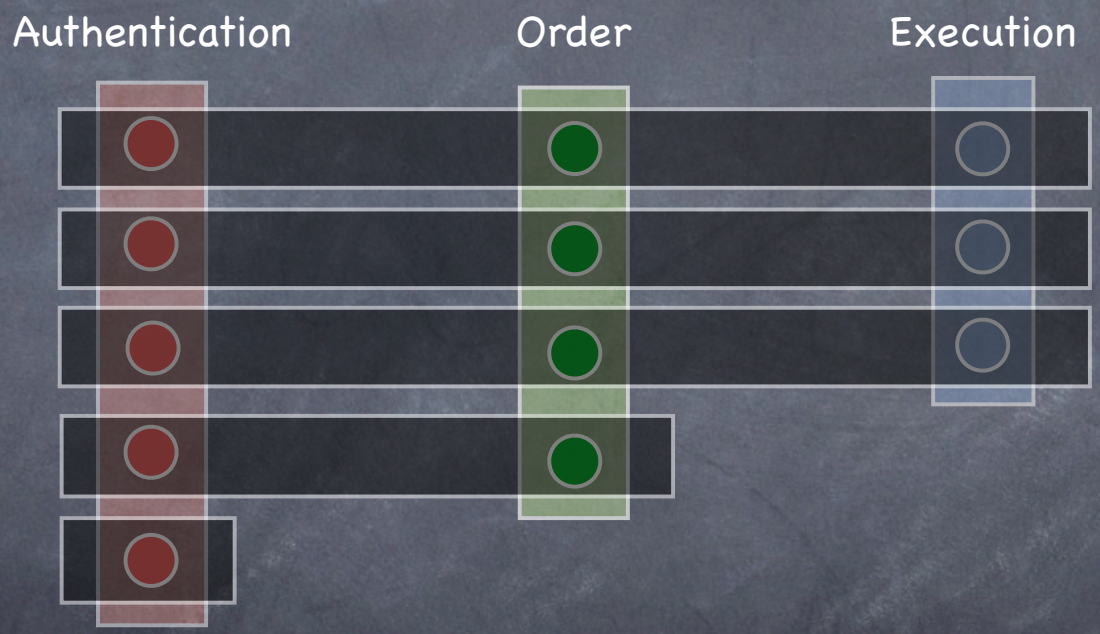


Faulty Primary

A More Perfect Separation

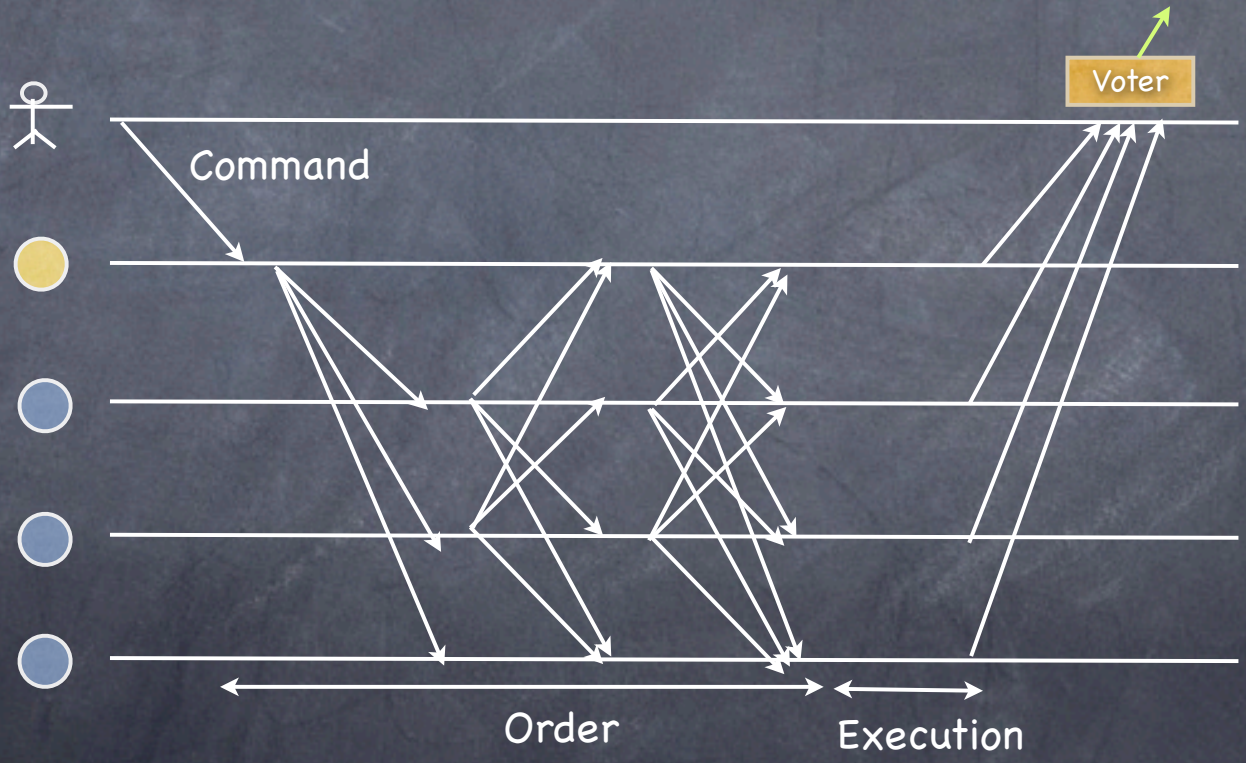


A More Perfect Union



Speculating

- Zyzzyva (Kotla et al 2003)



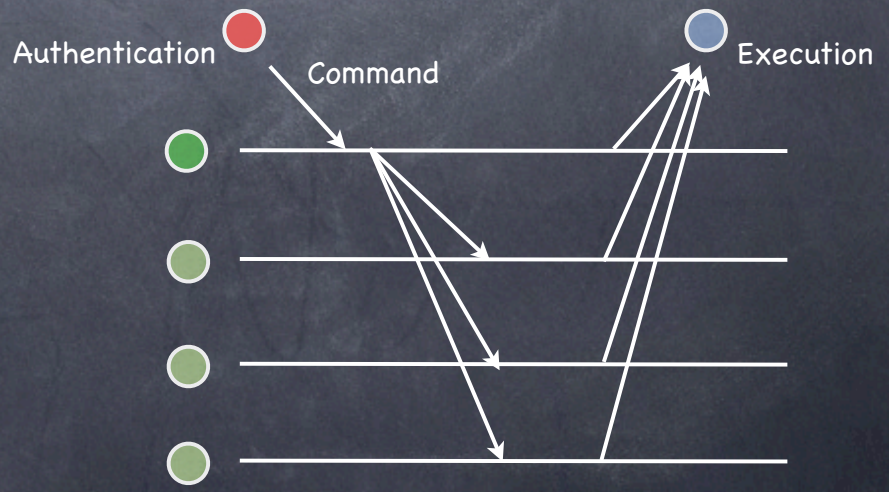
Misunderspeculation

- Speculation is a good idea
- Speculative execution is a bad idea
  - if wrong, lots of work
  - it's not about execution, anyway

Misunderspeculation

- Speculation is a good idea
- Speculative execution is a bad idea
  - if wrong, lots of work
  - it's not about execution, anyway

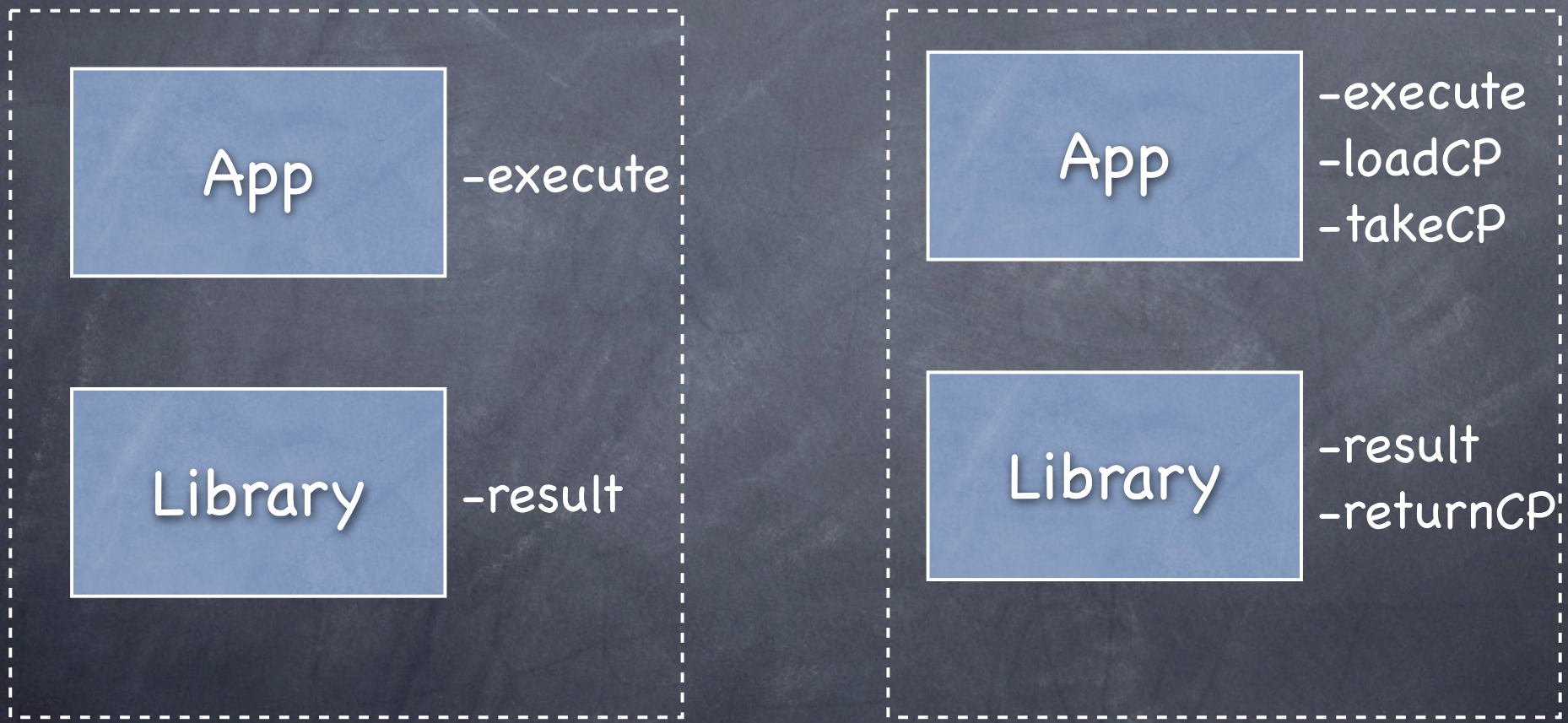
- UpRight
  - speculative ordering
  - execution nodes never roll back



Revisiting conventional wisdom

- Failure model
- BFT implementation
- API

API



Old World Order

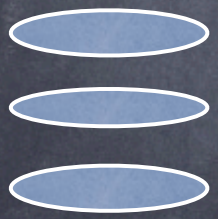
UpRight World Order

Case Study: Make HDFS UpRight

NameNode

Map files to blocks
Map blocks to data nodes

Users



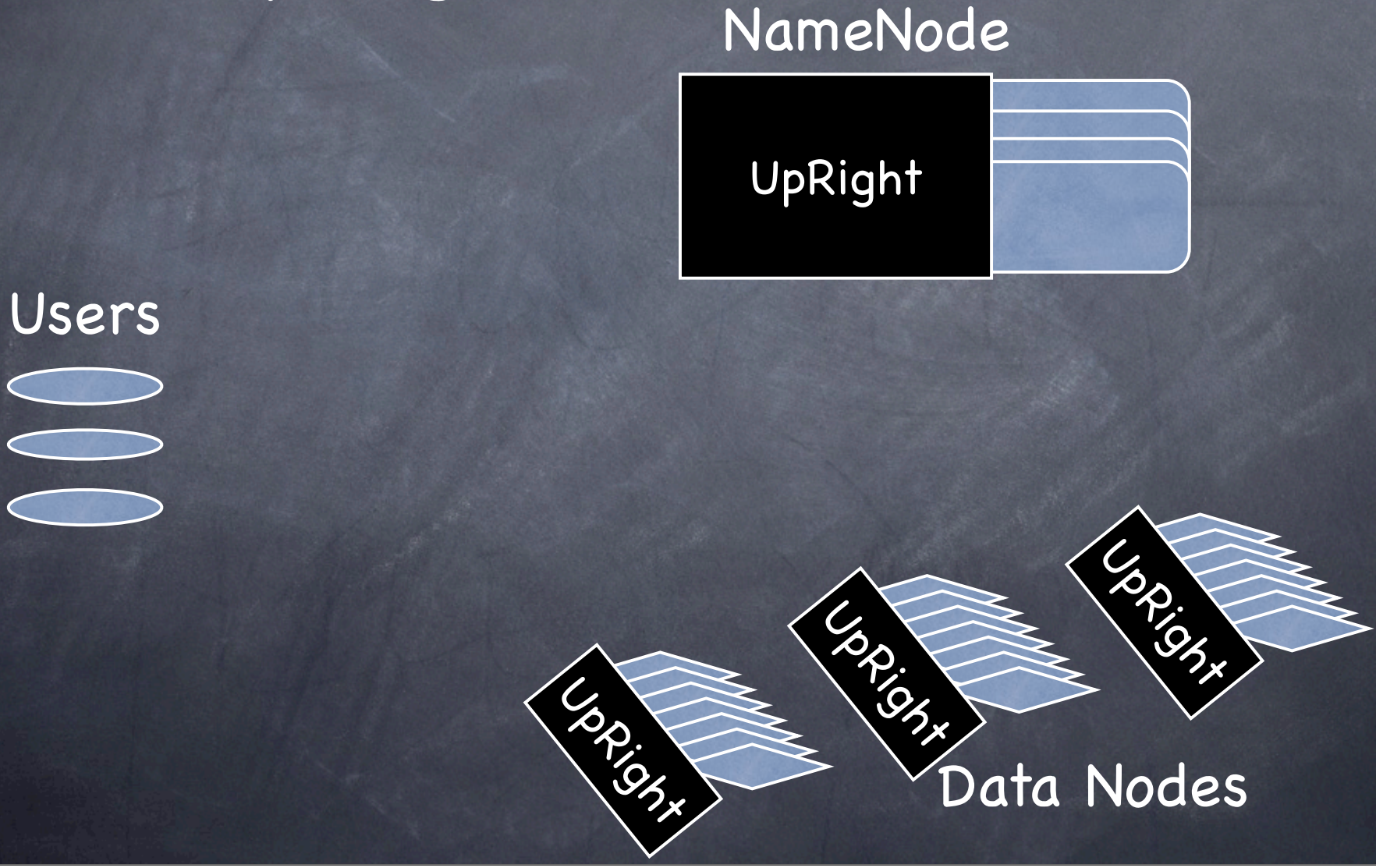
Data Nodes

Store blocks

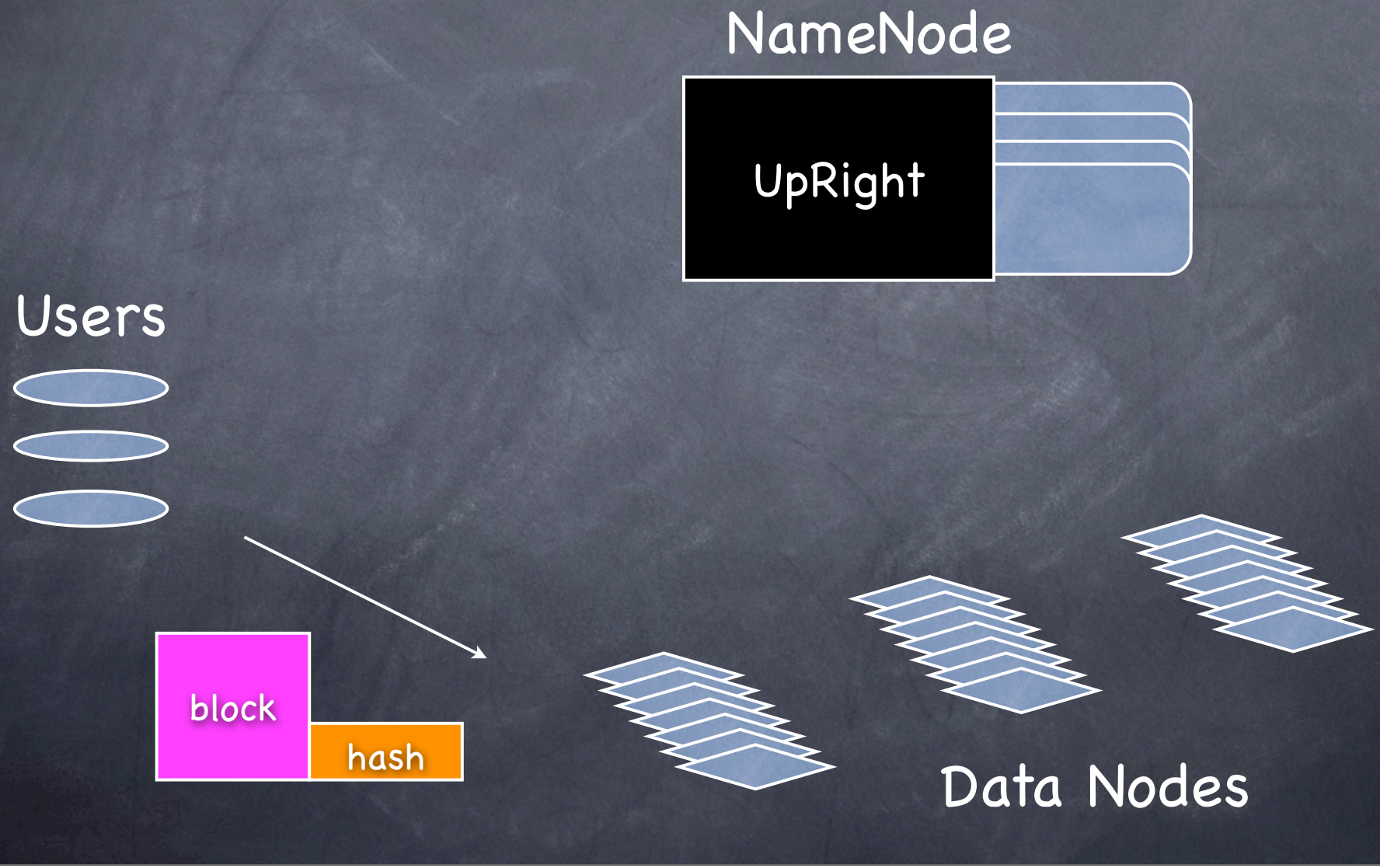
What was required?

- Make execution deterministic
  - ~150 lines of code
- Make checkpoints deterministic and complete
  - ~1500 lines of code
- That's it.

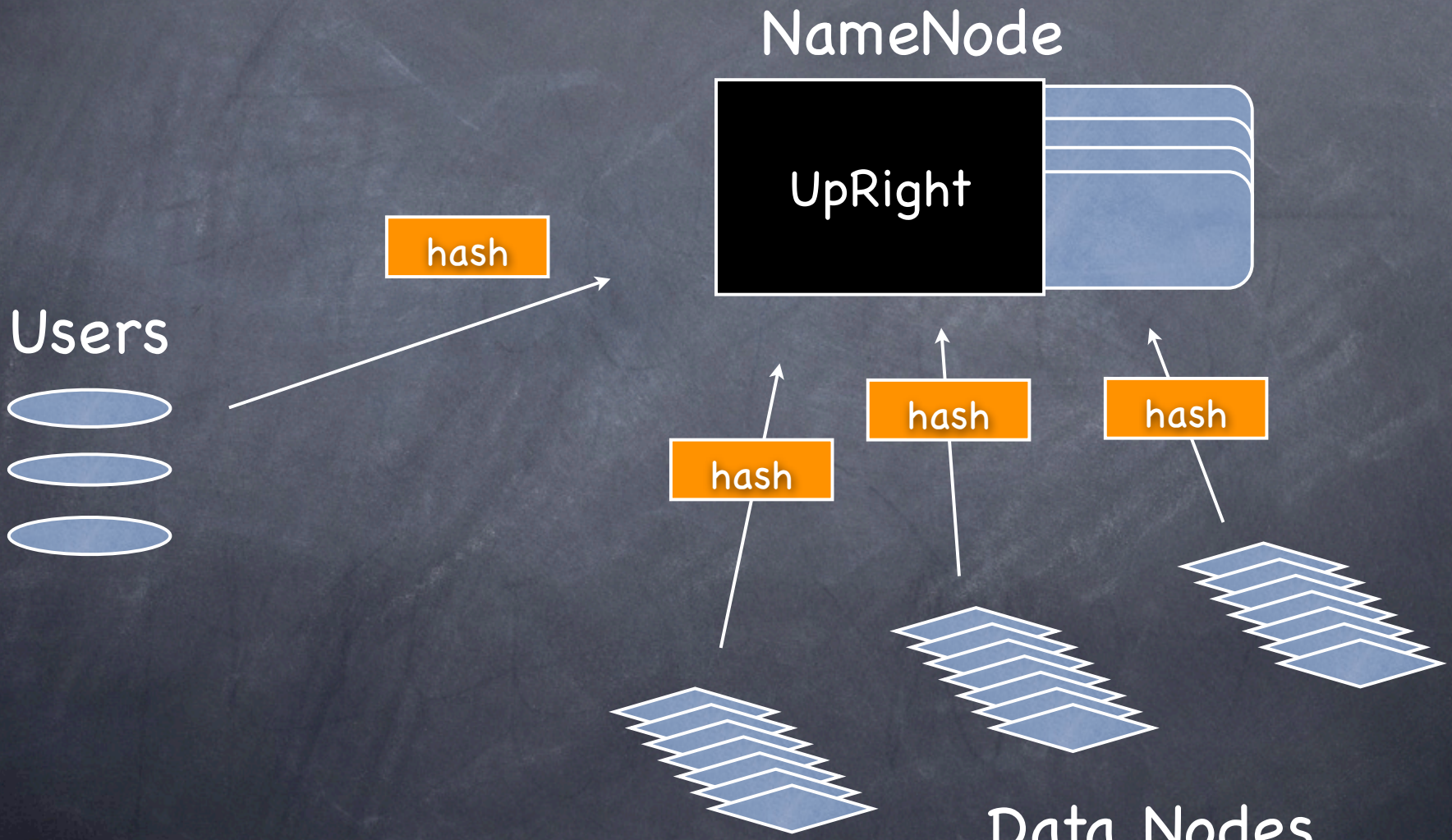
Do DataNodes Need the UpRight Treatment?



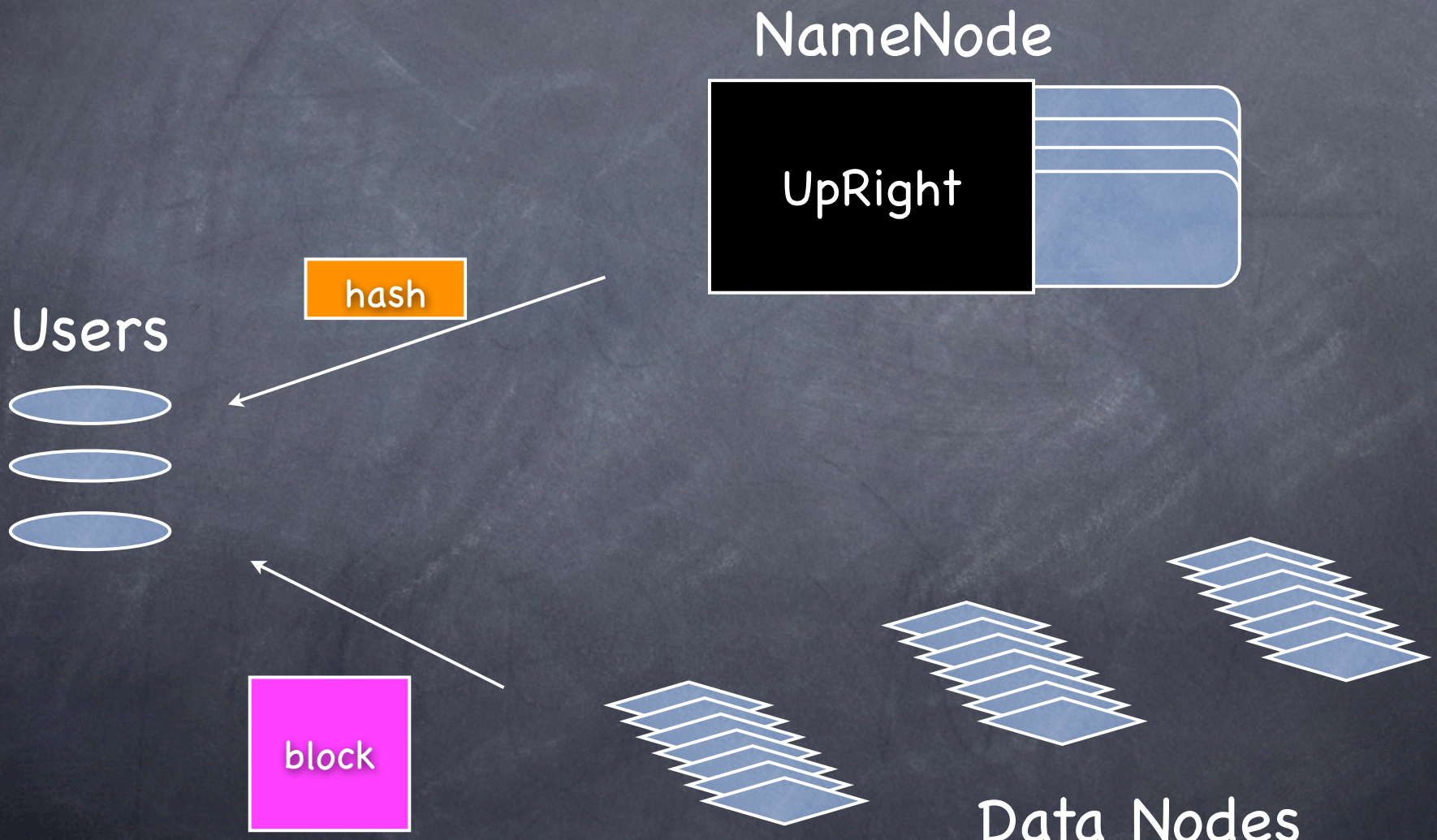
Modified DataNode



Modified DataNode



Modified DataNode



HDFS LOC Changes

NameNode Execution	NameNode Checkpoints	DataNode Protocol
~150	~1500	~900

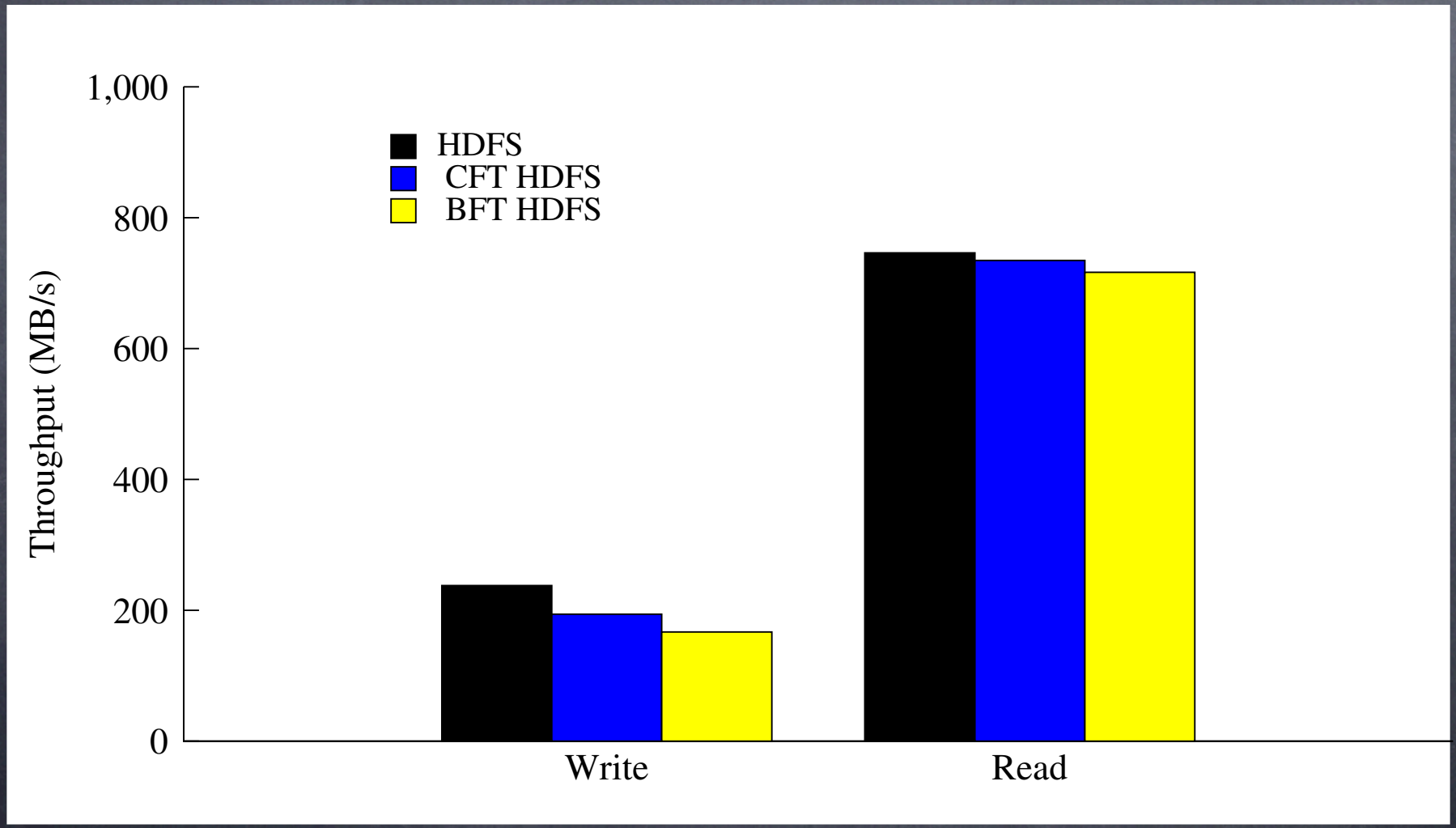
HDFS: ~37k LOC total

HDFS Evaluation

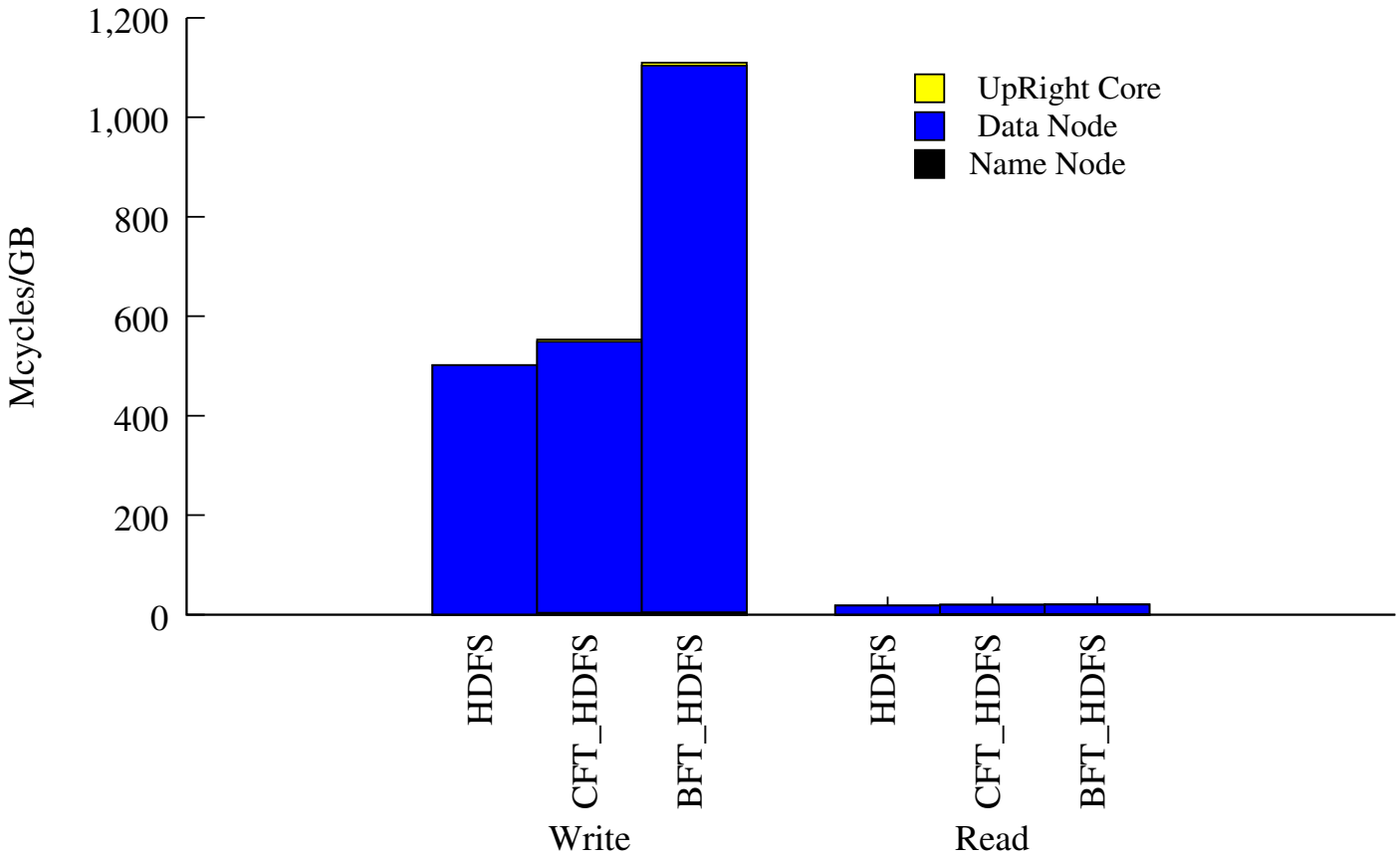
- Amazon S3 small instances
- 50 clients
  - each client writes/reads 1 GB file
- 50 data nodes

HDFS configuration	Authentication / Order / NameNodes	DataNode replication factor
Original HDFS	- / - / 1	3
CFT HDFS (u=1,r=0)	3 / 3 / 3	3
BFT HDFS (u=1,r=1)	4 / 4 / 3	3

HDFS Throughput



HDFS Computational Costs



This talk

- BFT in real systems
  - ZooKeeper, **H**adoop **D**istributed **F**ile **S**ystem
- What it took
  - UpRight
  - BFT Implementation
  - API

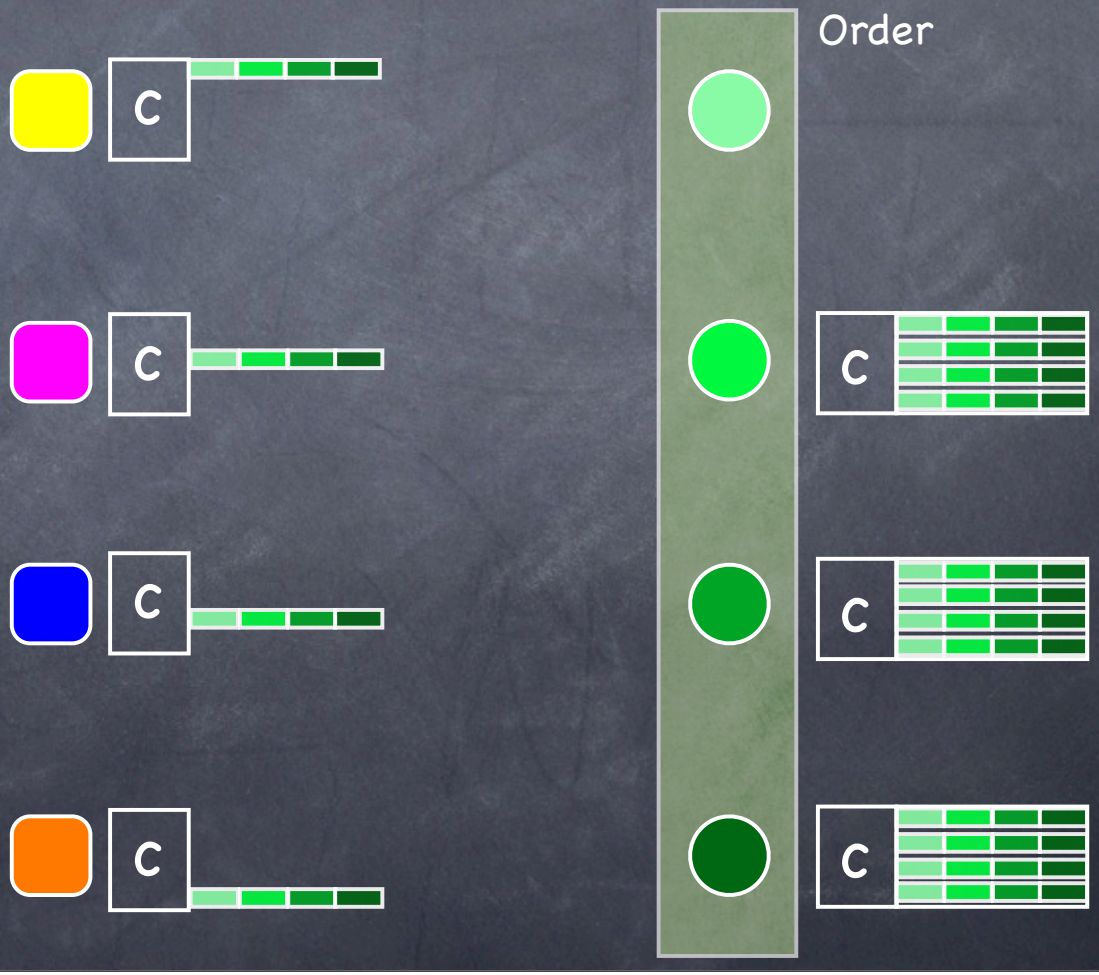
What the future holds

- The plural of "anecdote" is not "data"
  - Quantify the risks
    - how frequently do Byzantine failures occur?
    - how much damage can they do?
  - Quantify the benefits
    - what fraction of these failures does BFT mask?

Matrix signatures

(Aiyer et al 2008)

- Separate order from authentication



Matrix signatures

(Aiyer et al 2008)

Matrix signatures

(Aiyer et al 2008)

Primary orders request if sufficiently many valid MACs

Matrix signatures

(Aiyer et al 2008)

- Validity: request is from client
  - $n \geq r + 1$

Primary orders request if sufficiently many valid MACs

Matrix signatures

(Aiyer et al 2008)

- Validity: request is from client
  - $n \geq r + 1$
- Transitive validity: convince others
  - $n \geq 2r + 1$

Primary orders request if sufficiently many valid MACs

Matrix signatures

(Aiyer et al 2008)

- Validity: request is from client
  - $n \geq r + 1$
- Transitive validity: convince others
  - $n \geq 2r + 1$
- Liveness: request will go through
  - $n \geq 2r + u + 1$

Primary orders request if sufficiently many valid MACs